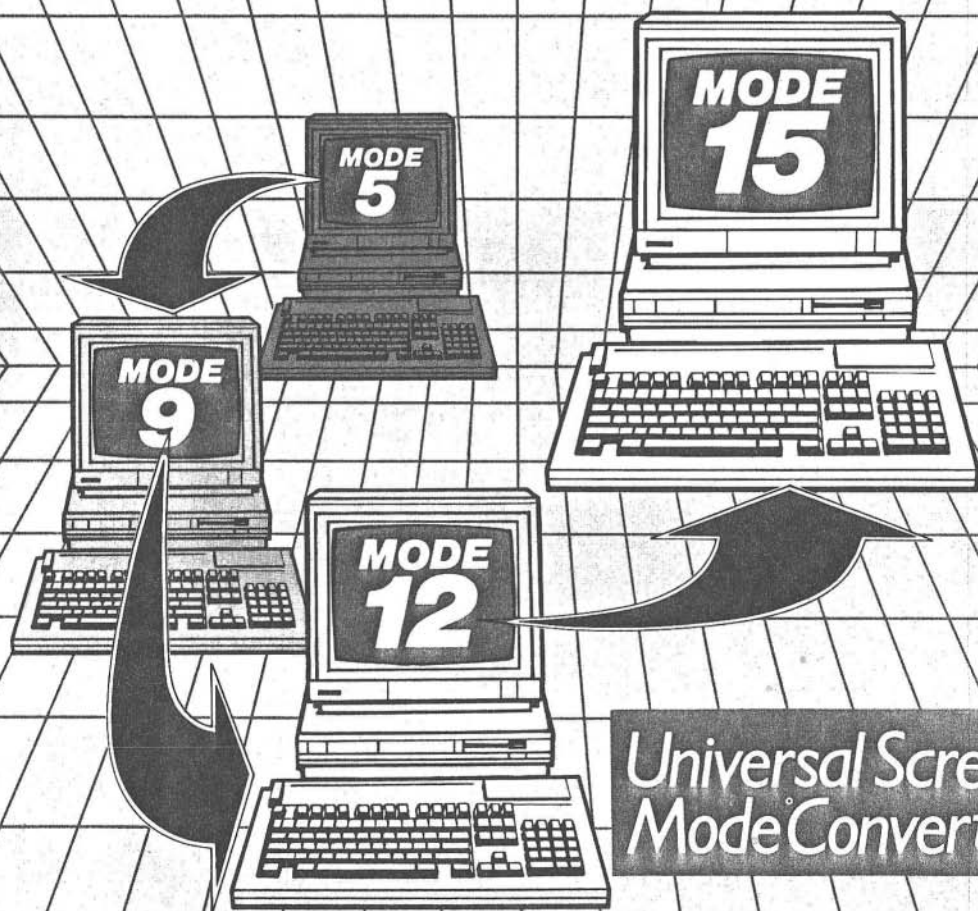
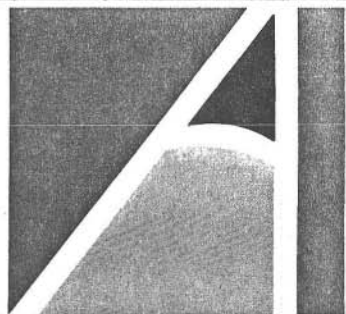


Volume 2  
Issue 6

May  
1989

Price £1.20

# RISC USER



THE MAGAZINE AND SUPPORT GROUP  
EXCLUSIVELY FOR USERS OF THE ARCHIMEDES

# RISC USER

## CONTENTS

### FEATURES

News	4
RISC OS Sprite Calls	13
ARC Procedure Library	19
Archimedes Visuals:	
Spinning Text	23
Writing a Machine Code Utility	31
RISC OS Basic V	34
Function Key Codes	40
Postbag	53
Technical Queries	55
Hints & Tips	57

### UTILITIES AND APPLICATIONS

A Partial Renumber Utility	9
Universal Screen Mode Converter	17
Movie Maker Stand-alone Display	27
The RISC User Notepad (Part 2)	47

### REVIEWS

Render Bender	6
Designer Intro	28
Premier Text Processing	37
Which Assembler?	43

RISC User is published by BEEBUG Ltd.

Co-Editors: Mike Williams, Dr Lee Calcraft

Assistant Editor: Kristina Lucas

Technical Editor: David Spencer

Technical Assistant: Glynn Clements

Production assistant: Sheila Stoneman

Advertising: Sarah Shrive

Subscriptions: Mandy Mileham

BEEBUG Ltd (c) 1989

All rights reserved. No part of this publication may be reproduced without prior written permission of the Publisher. The Publisher cannot accept any responsibility whatsoever for errors in articles, programs, or advertisements published. The opinions expressed on the pages of this journal are those of the authors and do not necessarily represent those of the Publisher, BEEBUG Limited.

BEEBUG, Dolphin Place, Holywell Hill, St Albans,  
Herts AL1 1EX. Tel. St Albans (0727) 40303



## The Archimedes Magazine and Support Group.

### EDITORIAL

Taking the good news first, the RISC OS upgrade sets, complete with manuals and ROMs have begun to come through from Acorn, as of early April. The not-so-good news is that the rate of supply is considerably less than the overwhelming demand for the product. This means that if you did not order your upgrade early, it will be well into May, and maybe even beyond until you receive your upgrade.

Acorn are using three Japanese suppliers, including Fujitsu and Hitachi, with a combined rate of delivery of around 4000 per month. The accompanying discs and manuals are all ready to go, and Acorn are shipping the upgrade sets as soon as they receive each ROM delivery.

As you may have gathered from recent assertions in the computer press (including RISC User), and counter denials from Acorn, something is brewing at the Cambridge Technopark. And announcements about a new RISC machine are expected shortly, though we are not permitted to disclose any details about it at this stage. Associated with this new development, Acorn are releasing a revamped 400 series Archimedes with a much lower entry price.

The 400/1 series will have two models, the 410 and the 420. Both will run 10% faster than previous machines, and will be supplied with 4 internal module sockets. The 410 will have 1 Mbyte of RAM upgradable to 2 or 4 Mbytes, and will have a hard disc interface. The 420 will have 2 Mbytes of RAM upgradable to 4 Mbytes, and will be supplied with an internal 20 Mbyte hard disc. The ex VAT prices for the two machines will be £1199 and £1699 respectively.

Finally to much more mundane matters. This issue of RISC User sees a new Technical Queries page carrying answers to selected postal queries. If you find this worthwhile, please let us know.

*This month's telesoftware password is **commando**  
(See BEEBUG pages on Micronet)*

## ARCHIMEDES EXTRAS

Nearly two years after it was promised, Acorn have finally launched the Archimedes 410. Indeed, they have also launched a 420 and an updated 440, forming a new 400 series which many people believe will replace the current 305 and 310. The new machines, officially called the 410/1, 420/1 and 440/1 and the 410/1 and 420/1 are fully upgradable to the 440/1 and all include RISC OS as standard. The table below summarises the features.

Machine	RAM (Mb)	Four-slot backplane	Hard disc controller	Hard disc fitted	Price
410/1	1	Yes	Yes	No	£1378.85
420/1	2	Yes	Yes	Yes (20Mb)	£1953.85
440	4	Yes	Yes	Yes (50Mb)	£2873.85

The prices given are VAT inclusive, and are for the base system. A further £253 should be added for a colour monitor.

## PRICES UP

To coincide with the launch of the 400/1 series, Acorn have announced a price increase for the 310. The base price for the 310 rises from £960.25 to £1033.85 (all prices inc. VAT), with a colour monitor again adding another £253. The price of the Master 128 computer also rises, to £504.85. However, for the first time, Acorn have introduced discounted prices for educational establishments. The recommended price for a 310 is £720 (ex VAT).

BEEBUG will continue to supply Archimedes machines at the old price for as long as possible, and will still include the package of free extras (see retail catalogue for details). Furthermore, BEEBUG will match any bona fide deal offered by any Acorn dealer.

## CRYSTAL PRINT

Those considering purchasing a PostScript compatible laser printer for the Archimedes might like to consider the Qume Crystalprint Publisher as an alternative. This is not a laser printer, but instead uses a new form of liquid crystal shutter in conjunction with a normal light source.

The Crystalprint Publisher comes with 3Mbytes of memory and 11 built-in fonts, with the option to download more when needed. The printer costs £3449 (inc. VAT), which may not seem very cheap, but it is in fact the cheapest PostScript printer available. More details from Qume, Qume House, Parkway, Newbury, Berkshire RG13 1EE, tel. (0635) 523200.

## SERIAL CARD

Intelligent Interfaces, the company known for making specialised hardware add-ons, has launched a new serial podule for the Archimedes. The single width board offers two RS423 ports, both of which are fully compatible with the one fitted as standard (except for the bugs). Each port offers totally independent data format and baud

rates (75 to 19200). The software supplied in ROM on the podule allows the new ports to be accessed by the operating system directly. The RS423 podule costs £224.25 (inc. VAT). More details from Intelligent Interfaces, 43b Wood Street, Stratford Upon Avon, Warwickshire CV37 6JQ.

## LOGIC DESIGN

For any Archimedes owners wishing to design their own hardware, Atomwide has just released a PAL and PROM programmer. This allows PROMs (Programmable ROMs) and PALs (Programmable Array Logic) to be programmed from data specified in standard JEDEC files. The programmer consists of an Archimedes podule, a programming socket, and a power supply. The whole setup costs £199 (inc. VAT) and should prove very useful for developers wishing to design their own Archimedes podules. More details from Atomwide Ltd., 23 The Greenway, Orpington, Kent BR2 2AY, tel. (0689) 38852.

## MULTI-PIN PRINTING

Epson has launched the world's first 48-pin dot matrix printer. The TLQ4800, as it is called, offers a printing speed of 250 characters per second, two letter quality fonts, a wide range of type styles, and full programmability

# NewsNewsNews NewsNew

via the control panel. Epson says that the TLQ4800 offers as good a quality as a laser printer, but at £2528.85 (inc. VAT) it should do. More details from Epson UK, Campus 100, Maryland Avenue, Hemel Hempstead, Hertfordshire HP2 7EZ.

## SHOW STORIES

It now appears certain that there will not be a Micro User show in London this year. Therefore, the only Acorn specific exhibition this year will be the Acorn User show at Alexandra Palace from the 21st to the 23rd of July. BEEBUG and RISC User will be there on stand 55.

## DOUBLE SAMPLES

Armadillo Systems have launched an enhanced version of their Armadillo sound sampler. The new device offers sixteen-bit sampling, which gives a dynamic range comparable to that of a compact disc. This new sampler is supplied with a software package called HighNote, which Armadillo claims is designed to make the professional musician feel at home with the computer. The whole package costs £1380 (inc. VAT), with much of this cost being attributed to the analogue to digital converter chip.

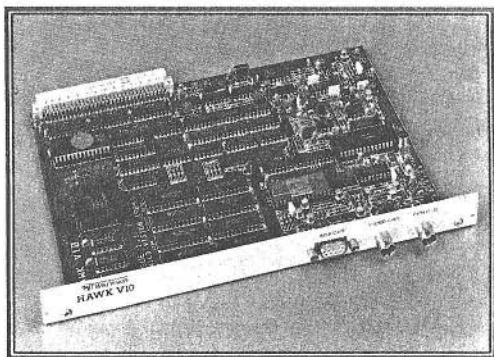
Armadillo has also extended its range of eight-bit samplers to include stereo and MIDI (Musical Instrument Digital Interface) versions. The eight-bit range starts at £155.25. More details can be obtained from Armadillo Systems Ltd., 17 Glaston Road, Uppingham, Leicestershire LE15 9PX, tel. (0572) 822499.

## MAKING AN IMPRESSION

The long awaited desktop publishing (DTP) package form Computer Concepts is set to be released in June. The system, called Impression, is designed to create a complete page from start to finish. This means that the package can be used as a word processor to enter and edit text, and as a page layout system to adjust the final appearance of the page. Impression will cost around £150, and details of availability can be obtained from Computer Concepts, Gaddesden Place, Hemel Hempstead, Hertfordshire HP2 6EX.

## IMAGE PROCESSING

The Hawk V10, from Wild Vision, is a professional image processing system for the Archimedes. The system consists of a full-width podule (pictured) which can capture up to four separate images in 256 grey shades, with a resolution of 256 by 256. Up to fifty frames a second may be grabbed, giving a continuous smooth display, and colour images can be handled by grabbing the red, green and blue components separately. Unlike other systems, the images are stored in memory on the podule, and are displayed using a separate monitor. This allows the full power of the Archimedes to be used for processing the images in real-time. The sample software supplied with the system provides a number of image processing functions, including a Convolver to enhance the focus of an image. The Hawk V10 starts from £747.50 (inc. VAT). More details from Wild Vision Ltd., 6 Jesmond Road, Newcastle Upon Tyne NE2 4PQ, tel. 091-281 8481.



The Hawk v10 expansion card

## WORLD DEVELOPMENT DATABASE

Geoscan, written by Limmasol Grammar School, is a learning resource for the GCSE exams in Geography and Economics. Geoscan offers a database covering locational, climatic, physical and socio-economic information on over 100 countries. Students can retrieve the information in a variety of forms, including diagrams and graphs. A number of statistical operations can also be performed on the data, allowing comparisons to be made between countries. Geoscan costs £45 (inc. VAT), and is available from Passkey Marketing, PO Box 649, Shenley Lodge, Milton Keynes MK5 7AX.

RU

David Spencer looks at Clares' new 3D ray tracing package.

Anyone who experimented with Roger Wilson's ray tracing program from last month's RISC User will appreciate how powerful a technique it is. However, modifying the program for different images is far from simple. Render Bender from Clares is a new package designed to take the pain out of ray tracing and enable advanced effects to be achieved with relative ease. For anyone who did not read last month's article, ray tracing is basically a brute force technique for determining the appearance of a 3D image made up of solid objects. The technique works by considering every pixel on the screen, and conceptually tracing rays of light back around all the objects until they reach the original light source.

## CREATING A SCENE

Before continuing, I must make one thing clear - Render Bender is not, and does not claim to be, a drawing or painting package. In fact, Render Bender offers no facilities for interactively editing a picture on screen. Instead, Render Bender is designed solely for the creation and animation of scenes consisting of geometric objects.

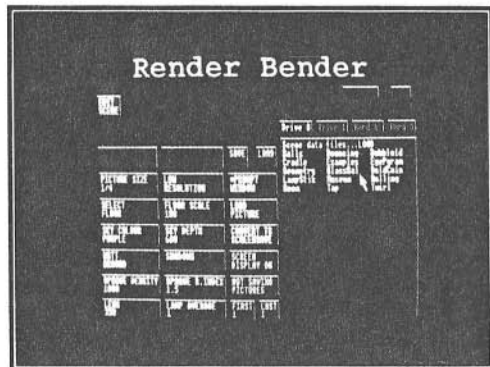
The process of creating and displaying an animated sequence consists of several stages:

1. The objects making up the scene are defined using a special programming language, together with the changes from frame to frame (if any) to produce an animated sequence.
2. The text description of the scene is compiled into Render Bender's internal format.
3. The individual frames are ray traced according to a number of options which may be set up.
4. The resultant frames are compressed using a deltafile method, and a display program incorporated.
5. Finally, the completed sequence is displayed.

## RUNNING RENDER BENDER

When you start the Render Bender disc you are presented with a very swish title screen consisting of a rotating gold coin. This gives

way to an equally impressive selection screen which allows you to choose between the editor and ray tracer program, or the deltafile animator. Both of these screens were created using the package itself.

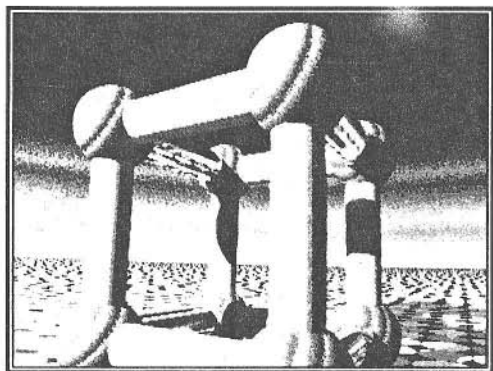


Render Bender menu screen

The stages of producing an animation sequence are performed by two independent parts of the program, linked by a selection screen. The first of these is the editor and ray tracer. This allows you to create a scene with the scene description language, and ray trace it to create a number of frames.

## THE SCENE DESCRIPTION LANGUAGE

The scene description language is the key to creating a picture. The format is entirely text-based, and consists of a series of keywords followed by a number of qualifiers and parameters, all of which must be separated by commas. There are basically three types of commands, these being used to position light sources, the viewer, and any objects. All positions are specified in 3-space as X, Y and Z co-ordinates, with the origin effectively at the centre of the image. The view point for the scene is given by the viewer's position, and either the point they are looking at or the direction they are looking in. There are three possible types of light source - a ray, a point source, and a spot-light, and these can be mixed at will in a scene.



Sample ray-traced image

For a spotlight you can control not only the direction of the beam, but also the angle through which it spreads. Objects in the scene can be chosen from pyramids, boxes, tubes, discs spheres and polygons, and are specified in terms of their position, dimensions and orientation. Additionally, each object can be given a colour, and a surface attribute which determines whether it is Shiny, mirror-like, metallic, opaque or bulb-like. (A bulb appears brilliant white, regardless of the light falling on it).

Animation is achieved by incorporating variables in the definitions. The ray tracer will automatically update these variables in the specified way between frames.

Once a scene description has been written, it has to be compiled from within the editor. This only takes a few second, and then the scene is ready to be ray traced. Before this is done, a number of parameters need to be set up. For example, you can change the floor pattern, floor scaling, sky colour, viewing lens, opacity of opaque objects, the refractive index of objects that transmit light, and shadowing. You can also select whether to ray trace in mode 13 or 15, and whether to scale the picture for a faster preview. The actual ray tracing can be quite time consuming. For a complex, full screen image in mode 15 with shadowing, it can take about an hour a frame. In other words, a fifty frame sequence could take two days!

A single ray traced picture can be exported directly from the ray tracer. However, an animation sequence must be processed by the deltafile animator. This part of Render Bender takes a sequence of picture files, and compresses them to produce a final sequence file. The order in which the frames are displayed is also determined at this point. The final sequence can either be displayed directly from within Render Bender, or saved and played back separately using a simple Basic program.

## PRESENTATION

Render Bender is supplied in an A4 size opening box, with the two discs slotted into the inside cover, and the manual recessed into the box. The box is printed in full colour and is very eye catching. This is a major departure from the 'video case' style box previously used by Clares, which I personally thought was always very drab. Also gone are the familiar Clares' disc labels, which have now been replaced by a more attractive design. My only quibble about the new presentation is that the review copy arrived somewhat squashed. Hopefully, Clares will improve their packaging to prevent this.

## DOCUMENTATION

The Render Bender manual has not escaped Clares revamp either. It is an A5 sized book, about 80 pages long, and printed on high quality paper with glossy full colour covers. The cheap looking plastic comb binding has been replaced with genuine Wire-O binding. The manual is properly typeset, and is laid out in such a way as to make it very impressive and easy to read. There are plenty of clear illustrations, and several direct screen dumps.

The manual starts off with a brief overview of the package and the principle of ray tracing, and then continues with an exhaustive explanation of how to start Render Bender and how to install it onto a hard disc. Render Bender is fully RISC OS compatible, and instructions are given for starting it from the Desktop.

The next section covers the use of the Editor and Ray tracer. This is presented in such a way that a first-time user can get the system



up and running without getting bogged down in all the various controls. The final main section covers the Animator, and is again easy to follow for the first-time user. This section also explains how to display animated sequences without Render Bender being present, and how to use the Animator to animate screens generated by other packages.

Two appendices explain the syntax of the all-important scene description language, and any possible errors that may occur while running Render Bender. The manual is rounded off with a comprehensive index. The only fault I could find with the manual is the lack of a detailed explanation of ray tracing. Admittedly, it does point out that this would require an entire book, but Clares could still have recommended one. Personally, I like the book suggested last month.

## CONCLUSION

Clares have once again come up with a professional product, and also one which does not

suffer from the problem of highly graphical, but totally confusing, icons - a criticism of Clares' products in the past. There is no doubt that Render Bender is excellent at performing its function, and is a very well thought out package. It ensures that all that is needed to produce stunning animations is a good imagination. But what use is it? The Archimedes has now been around for nearly two years, and the thrills of stunning images and animation sequences are, I feel, rapidly subsiding. I don't think many people will buy Render Bender just to draw pretty pictures and sit looking at them for hours, especially at £80.

Product:	Render Bender
Supplier:	Clares Micro Supplies, 98 Middlewich Road, Rudheath, Northwich, Cheshire CW9 7DA. Tel. (0606) 48512
Price:	£79.95 inclusive

*This month's magazine disc contains some sample frames produced using Render Bender.* **RU**

## ARCHIMEDES SOFTWARE

- Disc 1 EMACS. High powered editor with built in language.
- Disc 2 Microspell. 43,000 word spelling checker for EMACS or standalone use.
- Disc 3 Fortune Cookie over 7,000 amusing quotations.
- Disc 4 XLISP. object oriented version of LISP with C source.
- Disc 5 C Toolkit. Over 20 programs grep, awk, sed, ed, make, tail, cross ref. pretty print.
- Disc 6 Kermit. Archimedes, BBC & MSDOS versions of file transfer prog.
- Disc 7 WIMP chess. Good standard wimp based chess program with lots of features.
- Disc 8 Cross Star. Wimp based crossword solver with huge dictionary.
- Disc 9 File Tools arc, compress, uuencode, cut, paste, fgrep, rotate, strings.
- Disc 10 More C Tools. yacc and lex with examples and C source. diff, ctags etc.
- Disc 11 Little SmallTalk. Nice implementation of the original object oriented language. Includes C source, SmallTalk code and documentation.
- Disc 12 The World Digitised. 130,000 coordinates of points on the Earth.
- Disc 13 Birdlog. Database and statistics package for birdwatchers.
- Disc 14 Gnu Tools#1. High quality UNIX tools with C source. egrep, grep, awk, diff, sed.
- Disc 15 Clip Art#1. Over 50 pictures to paste into DTP/ArcDraw etc files.

***Each disc is £5.99 inc. Buy four claim one free!***

**David Pilling, P.O. Box 22, Thornton Cleveleys, Blackpool. FY5 1LR**

*No surcharge on overseas orders. More details available on request.*



# A PARTIAL RENUMBER UTILITY

David Spencer offers a utility to tidy up your Basic programs.

One very useful function that is not even possible using the Basic Editor is to be able to renumber only part of a Basic program. For example, you may wish to open a gap in the program to insert a few extra lines, or you may have adopted a numbering scheme where, say, each procedure started at a line which was a multiple of 1000. The machine code utility given here implements just such a renumber.

Start by entering the program given in the listing, and save it to disc. Running it will assemble a machine code utility (Arc utilities are discussed elsewhere in this issue), and save it with the filename 'Renumber'.

## COMMAND SYNTAX

The Renumber command can be used in one of three forms:

```
*Renumber -Help
*Renumber -Check [-Start <lnum>]
[-End <lnum>]
*Renumber [-NoConfirm] [-Start
<lnum>] [-End <lnum>] [-First <lnum>]
[-Increment <lnum>]
```

Any of the parameters enclosed in square brackets may be omitted, in which case various defaults will be assumed, these being detailed later. It is also normally possible to omit the keywords (for example putting 12000 instead of -END 12000). However, the keywords must be used when omitting a parameter would lead to confusion over the meanings of the others. For example:

```
*Renumber 10 100 1
would be the same as:
```

```
*Renumber -Start 10 -End 100 -First 1
Had it been intended that the '1' was the
Increment, the 'First' parameter having been
omitted, then the correct form would be:
```

```
*Renumber 10 100 -Increment 1
In any case, all the keywords can be abbreviated
to just the first letter, so the above command
would become:
```

```
*Renumber 10 100 -I 1
```

Incidentally, whenever a numeric parameter is required, an expression yielding a numeric result can be used instead.

## USING THE COMMANDS

The first form of the command will simply print the syntax and exit immediately:

```
*Renumber -Help
```

The second form of the command will check that the line numbers of a program are in sequence. This is because by using a partial renumber, it is possible to end up with a situation where one line has a line number equal to, or less than, the previous line. This will not normally affect the execution of a program, but will confuse commands which operate on a range of lines (for example LIST and DELETE). By using:

```
*Renumber -Check
```

all discrepancies in the line sequencing will be displayed in the form:

```
xxxx - yyyy
```

where xxxx is the first line number, and yyyy is the second. By specifying start and end line numbers, the check can be restricted to just a part of the program, for example:

```
*Renumber -Check 100 1000
```

If the start line number is omitted, the first line of the program is assumed, and if the end is omitted, the last line is assumed. If the specified lines do not exist, then the numbers are treated in the same way as for LIST.

The third form of the command is the one actually used for renumbering. As shown in the syntax above, there can be up to five parameters. The first of these is an optional qualifier (-NoConfirm), and controls what action is taken when renumbering would result in out of sequence line numbers occurring. If the qualifier is not given, then you are asked to confirm that you don't mind out of sequence numbers. On the other hand, if the qualifier is given then no confirmation is sought, although a warning will be given. The second and third parameters specify the start and end of the range to be renumbered, as above. The fourth parameter is the starting line number for the new numbering, and the final one is the increment to be used when renumbering. Both these default to ten if omitted.



The following examples should make the syntax clearer:

**\*RENUMBER**

Renumber whole program starting at line ten, in increments of ten (just as with Basic's RENUMBER).

**\*RENUMBER 100 200 1000**

Renumber lines between 100 and 200. Start at 1000 and increase in increments of 10.

**\*RENUMBER -NoConfirm 100 200 1000 50**

As above, but use an increment of 50. Additionally, don't seek confirmation if out of sequence numbers result.

**\*RENUMBER 80 -First 100**

Renumber all lines from line 80 to the end of the program to start at 100, and increase in increments of 10.

**\*RENUMBER -First 100 -Increment 25**

Renumber the entire program to start at line 100, and increase in increments of 25.

**\*RENUMBER -End 999 -First 1 -Increment 2**

Renumber all the lines up to line 999 to start at line 1 and go up in increments of 2.

```

10 REM > RenumSrc
20 REM Program   Partial Renumber
30 REM Version   A 1.00
40 REM Author    David Spencer
50 REM RISC User May 1989
60 REM Program   Subject to Copyright
70 :
80 DIM code 2000
90 FOR pass=0 TO 3 STEP 3
100 P%=code
110 [OPT pass
120 STMFDF R13!, {R1,R14}
130 MOV R0,#15:MOV R1,#0
140 SWI "OS_ChangeEnvironment"
150 MOV R6,R1:MOV R0,#18
160 ADR R1,basic:SWI "OS_Module"
170 CMP R6,R3:BCC notbasic
180 LDR R0,[R3,#-4]:ADD R3,R3,R0
190 CMP R6,R3:BCC getparam
200 .notbasic ADR R0,nberr
210 LDMFDF R13!, {R1,R14}

```

```

220 ORRS PC,R14,#1<<28
230 .getparam LDMFDF R13!, {R1}
240 ADR R0,defn:MOV R2,R12:MOV R3,#100
250 SWI "XOS_ReadArgs"
260 LDMVSFD R13!, {PC}
270 LDR R0,[R12]:CMP R0,#0
280 ADRNE R0,help:SWINE "OS_Write0"
290 LDMNEFD R13!, {PC}^
300 MOV R0,#&8F00:LDRB R0,[R0,#1]
310 CMP R0,#&FF:ADREQ R0,noprogram
320 LDMEQFDF R13!, {R14}
330 ORREQS PC,R14,#1<<28
340 BL getrange:LDR R0,[R12,#4]
350 CMP R0,#0:BEQ renumber
360 .seq LDRB R2,[R7,#1]
370 LDRB R1,[R7]:ADD R1,R2,R1,LSL #8
380 LDRB R2,[R7,#2]:ADD R7,R7,R2
390 LDRB R3,[R7]:LDRB R4,[R7,#1]
400 ADD R3,R4,R3,LSL #8
410 CMP R1,R3:BCC seqok
420 MOV R0,R1:MOV R1,#1:BL prtnum
430 SWI &120:SWI &100+ASC"-":SWI &120
440 MOV R0,R3:MOV R1,#0:BL prtnum
450 SWI "OS_NewLine"
460 .seqok CMP R7,R8:BNE seq
470 LDMFDF R13!, {PC}^
480 :
490 .renumber
500 LDR R6,[R12,#8]:MOV R9,R5
510 BL candg:LDR R0,[R12,#&14]
520 CMP R0,#0:MOVEQ R3,#10
530 BLNE gnum:MOVNE R3,R0
540 LDR R0,[R12,#&18]:CMP R0,#0
550 MOVEQ R4,#10:BLNE gnum:MOVNE R4,R0
560 CMN R9,#1:BEQ nof
570 CMP R9,R3:BCS badseq
580 .nof MOV R0,R7:MOV R9,R3
590 .schk LDRB R1,[R0,#2]
600 ADD R0,R0,R1:CMP R0,R8
610 ADDNE R9,R9,R4:BNE schk
620 LDRB R1,[R0,#1]:LDRB R2,[R0]
630 ADD R1,R1,R2,LSL #8
640 CMP R9,R1:BCC renum
650 .badseq CMP R6,#0:ADRNE R0,warn
660 SWINE "OS_Write0":BNE renum
670 ADR R0,query:SWI "OS_Write0"
680 MOV R0,#15:MOV R1,#1:SWI "OS_Byte"
690 SWI "XOS_ReadC":LDMVSFD R13!, {PC}
700 MOVCS R0,#ASC"N":BIC R0,R0,#&20
710 CMP R0,#ASC"Y":MOVNE R0,#ASC"N"

```

```

720 SWI "OS_WriteC":SWI "OS_NewLine"
730 LDMNEFD R13!, {PC}
740 .renum AND R0,R3,#&FF00
750 MOV R0,R0,LSR #8:STRB R0,[R7]
760 AND R0,R3,#&FF:STRB R0,[R7,#1]
770 LDRB R0,[R7,#2]:ADD R7,R7,R0
780 ADD R3,R3,R4:CMP R7,R8:BNE renum
790 BL lrefs:LDMFD R13!, {PC}^
800 :
810 .lrefs MOV R11,R14:MOV R9,R5
820 MOV R0,#&8F00:ADD R0,R0,#1
830 .lrefs2 LDRB R1,[R0]:CMP R1,#&FF
840 MOVEQ PC,R11:MOV R1,#3
850 .lrefs3 LDRB R2,[R0,R1]
860 ADD R1,R1,#1:CMP R2,#&8D:BLEQ ch
870 BEQ lrefs3:CMP R2,#&D:BNE lrefs3
880 LDRB R2,[R0,#2]:ADD R0,R0,R2
890 B lrefs2
900 :
910 .ch STMFD R13!, {R14}:ADD R2,R0,R1
920 LDRB R3,[R2]:LDRB R4,[R2,#1]
930 LDRB R5,[R2,#2]:MOV R3,R3,LSL #2
940 AND R6,R3,#&C0:EOR R4,R4,R6
950 MOV R3,R3,LSL #2:AND R6,R3,#&C0
960 EOR R5,R5,R6:ADD R5,R4,R5,LSL #8
970 MOV R3,#&8F00:ADD R3,R3,#1
980 MOV R4,R9
990 .hunt LDR R7,[R4],#4:CMP R7,R5
1000 BEQ hdone:LDRB R7,[R3,#2]
1010 ADD R3,R3,R7:LDR R7,[R4]
1020 CMP R7,#&FF00:BCC hunt
1030 STMFD R13!, {R0-R1}
1040 ADR R0,fail:SWI "OS_Write0"
1050 MOV R0,R5:MOV R1,#0:BL prtnum
1060 ADR R0,fail2:SWI "OS_Write0"
1070 LDR R0,[R13]:LDRB R1,[R0,#1]
1080 LDR R0,[R0]:ADD R0,R1,R0,LSL #8
1090 MOV R1,#0:BL prtnum
1100 SWI "OS_NewLine"
1110 LDMFD R13!, {R0-R1,R14}
1120 ADD R1,R1,#3:MOVS PC,R14
1130 .hdone LDRB R4,[R3,#1]
1140 LDRB R5,[R3]:ADD R3,R4,R5,LSL #8
1150 AND R4,R3,#&3F:ORR R4,R4,#&40
1160 STRB R4,[R2,#1]:AND R4,R3,#&3F00
1170 MOV R4,R4,LSR #8:ORR R4,R4,#&40
1180 STRB R4,[R2,#2]:AND R4,R3,#&C0
1190 MOV R4,R4,LSR #2:AND R5,R3,#&C000
1200 MOV R5,R5,LSR #12:ORR R4,R4,R5
1210 EOR R4,R4,#&54:STRB R4,[R2]

```

```

1220 ADD R1,R1,#3:LDMFD R13!, {PC}^
1230 LDMFD R13!, {PC}^
1240 :
1250 .basic EQUUS "BASIC":EQUB 0:ALIGN
1260 .nberr EQU0 0:EQUUS "Not in Basic"
1270 EQU0 0:ALIGN
1280 .noprog EQU0 0:EQUUS "No program"
1290 EQU0 0:ALIGN
1300 .help EQUUS "Renummer -Help"
1310 EQUW &D0A
1320 EQUUS "Renummer -Check [-Start <lno
>] [-End <lno>]":EQUW &D0A
1330 EQUUS "Renummer [-NoConfirm] [-Star
t <lno>] [-End <lno>] [-First <lno>] [-I
ncrement <lno>]":EQUW &D0A:EQUB 0:ALIGN
1340 .nerr EQU0 0:EQUUS "Bad line number
":EQUB 0:ALIGN
1350 .nrerr EQU0 0
1360 EQUUS "No room for renumber":EQUB 0
1370 ALIGN
1380 .bserr EQU0 0:EQUUS "Bad range"
1390 EQU0 0:ALIGN
1400 .defn EQUUS "Help/S,Check/S,NoConfi
rm/S,Start/E,End/E,First/E,Increment/E"
1410 EQU0 0:ALIGN
1420 .fail EQUUS "Failed with "
1430 EQU0 0:ALIGN
1440 .fail2 EQUUS " on line "
1450 EQU0 0:ALIGN
1460 .warn EQUUS "Line numbers will be o
ut of sequence":EQUW &D0A:EQUB 0:ALIGN
1470 .query EQUUS "This will result in o
ut of order line numbers - Continue (Y o
r N) ? ":EQUB 0
1480 :
1490 .gnum LDRB R1,[R0]:CMP R1,#0
1500 BNE bnum:LDRB R1,[R0,#3]
1510 LDRB R2,[R0,#4]:ORR R1,R1,R2
1520 BNE bnum:LDRB R1,[R0,#2]
1530 CMP R1,#&FF:BEQ bnum
1540 LDRB R0,[R0,#1]
1550 ADD R0,R0,R1,LSL #8:MOVS PC,R14
1560 .bnum ADR R0,nerr:LDMFD R13!, {R14}
1570 ORR PC,R14,#1<<28
1580 :
1590 .prtnum STMFD R13!, {R0-R3}
1600 MOV R3,R1:MOV R1,R12:MOV R2,#6
1610 SWI "XOS_ConvertCardinal2"
1620 CMP R3,#1:BNE prtnum2

```

Continued on page 18

We offer  
a unique  
inspection  
copy service

NOW  
AVAILABLE

Call for  
full details



# ARCHWAY

Have you  
spent long hours struggling  
with the programming complexities of the brilliant  
Archimedes **WIMPS** system?

With **ARCHWAY** that's all behind you. Now YOU can easily and quickly  
build multi-window programs with pop-up menus, icons, mouse control,  
etc. of a professional quality.

**ARCHWAY** provides tools, bricks, mortar and a basic structure. YOU slot  
the final bricks into place and add finishing touches.

A comprehensive suite of integrated tools lets you define and edit  
windows, menus, icons, dialogue boxes, mouse pointers, sprites, fill and  
line patterns, anti-aliased fonts, short cut keys and much more! You can  
import files (eg ARM machine code and music editor).

The bricks include an extensive library of functions and procedures  
providing ready access to many of Archimedes' most powerful features.

An in-depth user guide (300+ pages, ring bound) takes you gently step-  
by-step through a progressive series of program building examples with  
the emphasis on clarity.

The **ARCHWAY** run-time package together with  
script shells in BBC BASIC provide the structure.

**£79.95**  
incl. VAT & p/p

The complete system comes on 4\*800k discs (3 tools & 1 run-time). You  
need 1M of RAM to run the tools. One disc drive is sufficient.

**SPECIAL INTRODUCTORY OFFER:** No charge for upgrade to the  
extended RISC OS 2.0 version we will release in May 1989. (£99.95)



## SIMTRON

*Programs to  
help you*

**4 CLARENCE DRIVE, EAST GRINSTEAD,  
WEST SUSSEX RH19 4RZ Telephone (0342) 328188**



Cheques/POs/official orders or Access/Visa number and  
expiry date. 24-hour 'phone for Credit Card orders.





# RISC OS SPRITE CALLS

RISC OS provides the user with a powerful set of new sprite commands, as Lee Calcraff explains.

If you have seen the RISC OS Magnifying Glass in operation, then you will have seen something of the power of the new sprite calls available in RISC OS. The two that I am going to look at here allow you to scale a sprite up or down (as used by the Magnifying Glass demonstration), and to send VDU output directly to a sprite. By combining the two of these, you can produce a scaled output from a program within a window of any size. This trick has been used from the Desktop to allow Zarch to run within a small window.

- |    |                             |
|----|-----------------------------|
| 35 | Append a sprite             |
| 36 | Define pointer from sprite  |
| 50 | Plot scaled sprite mask     |
| 51 | Paint scaled character      |
| 52 | Plot scaled sprite          |
| 53 | Plot grey-scaled sprite     |
| 60 | Switch VDU output to sprite |
| 61 | Switch VDU output to mask   |

*Add 256 to these reason codes for named user sprites, or 512 for user sprites accessed by address.*

Table 1 The New Sprite Calls

## THE SPRITE CALLS

All of the new sprite calls use "OS\_SpriteOp". We took a look at this operating system call in Volume 2 Issue 3, when dealing with user sprites, and the reader is referred to that article for a "refresher" if need be. In what follows, I will be using the new calls with user sprites (since this is most appropriate for multi-tasking applications), and to start with, I will look at the code necessary to set up a sprite area, load a sprite file, and plot a sprite from it. All this can be achieved in just four lines:

```
10 DIM sp &2000:sp=&2000
20 sp!4=0:sp!8=16:sp!12=16
30 SYS "OS_SpriteOp",266,sp,"ICONS.Music"
40 SYS "OS_SpriteOp",290,sp,"Treble",6
40,512
```

As you can see, there is not much to it really. The first two lines set up a user sprite area of

&2000 (8K) in size, as described in the article referred to above. The first of the two OS calls loads a sprite file called "ICONS.Music" (supplied on the Arthur Welcome disc) into the area, and the second plots one of the sprites (called "Treble") from this file at co-ordinates 640, 512. The first numerical parameter in each case is the reason code for the operation, and the second the address of the sprite area, again as indicated in the previous article.

In fact the reason code for all sprite operations has two components, a number between 0 and 255 indicating the type of operation to be performed, plus a high byte indicating whether user or system sprites are in use, and in the former case, whether access is by name or address. The reason code to plot a sprite, used above, could thus be given as 34 (system sprites), 34+256 (named user sprites) or 34+512 (user sprites accessed by address). We have used the value 290 (i.e. 34+256), meaning that we are using user sprites, and that they will be accessed by name (the name "Treble" in this case). In what follows we shall continue to use this form of the call, referring to named sprites in a user area.

## SCALED SPRITES

To plot a scaled sprite, we use reason code 52+256 (i.e. 308), and we must set up four words in RAM to define the scale of magnification or reduction, and then supply a pointer to this 16-byte block in the call itself. The program in listing 1 achieves this, plotting the "Treble" sprite at co-ordinates 100, 100, at a horizontal magnification of 8 and a vertical magnification of 6. This is achieved using the 4 words of RAM at location *scale*. The first two numbers give the X and Y magnification factor, while the second two give the X and Y reduction factors. As you can see, the latter are currently set to 1. To reduce the horizontal size of the sprite by a factor of 2/3, we would use:

```
!scale=2
scale!8=3
```

and so on (i.e. multiply by 2 and divide by 3). As you will appreciate, this gives us very fine



control over the scale factor used for the sprite plot, and we can put this call to many uses. For example, we could make an image appear as a point on the screen, and then grow in size, while at the same time moving across the screen, by using a simple FOR loop.

#### Listing 1

```
10 REM >Ros1Sprt
20 REM Plot scaled sprite
30 :
40 MODE 12
50 DIM sp &2000:!sp=&2000
60 DIM scale 16
70 sp!4=0:sp!8=16:sp!12=16
80 SYS "OS_SpriteOp",266,sp,"ICONS.Mu
sic"
90 SYS "OS_SpriteOp",290,sp,"Treble",
640,512,0
100 :
110 !scale=8 :REM X multiply
120 scale!4=6 :REM Y multiply
130 scale!8=1 :REM X Divide
140 scale!12=1 :REM Y Divide
150 SYS "OS_SpriteOp",308,sp,"Treble",
100,100,0,scale
```

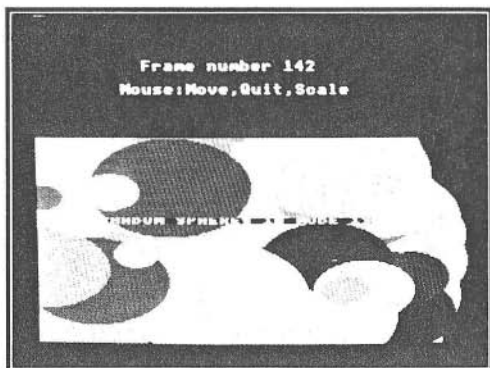
#### VDU OUTPUT TO A SPRITE

A call to "OS\_SpriteOp" with a reason code of 60+256 can be used to send all VDU output to a named sprite instead of to the screen. The sprite can then be displayed as and when required, using the calls already treated. The program in listing 2 achieves this. It first loads the Music sprite file, and plots the sprite named "Treble" as before. Next it performs Sprite Op 316 (i.e. 60+256) setting all VDU output to the "Treble" sprite. In practice text and graphics windows are set by this call to the size of destination sprite, with the graphics origin corresponding to the bottom left hand corner of the sprite, and screen output is sent to the sprite just as if it were a small windowed screen.

#### Listing 2

```
10 REM >Ros2Sprt
20 REM VDU output to sprite
30 :
40 MODE 12
50 DIM sp &2000:!sp=&2000
```

```
60 sp!4=0:sp!8=16:sp!12=16
70 :
80 SYS "OS_SpriteOp",266,sp,"ICONS.Mu
sic"
90 SYS "OS_SpriteOp",290,sp,"Treble",
640,512,0
100 :
110 REM VDU output to sprite
120 SYS "OS_SpriteOp",316,sp,"Treble"
TO R0,R1,R2,R3
130 CIRCLE FILL 20,40,16
140 :
150 REM Reinstate output
160 SYS "OS_SpriteOp",R0,R1,R2,R3
170 :
180 REM Plot it again
190 SYS "OS_SpriteOp",290,sp,"Treble",
768,640
```



#### Using the two sprite calls to send output from a program to a scaled window

Next the program draws a small filled circle at 20,40, which does not appear on the screen, but is sent to the sprite instead. Output is then reinstated (line 160) and the sprite re-plotted. Sure enough it appears with a white disc superimposed upon it.

To send VDU output to the Treble sprite, we have used just 3 parameters: the reason code, the pointer to the sprite area, and the sprite name, just as before. But now we store the register values on exit from the call in the four variables R0-R3, and these are used to reset VDU output after we have finished writing to the sprite. For this we use:



SYS "OS\_SpriteOp",R0,R1,R2,R3

These calls have been implemented by Acorn with a great deal of care, and all VDU variables are stored away and later reinstated when the calls are made. As a result the user does not need to keep track of window settings, origins, cursors and so on. It is all taken care of. Moreover, the VDU data sent to any sprite is given its own VDU queue, so that you can switch VDU output to and from sprites at any time.

To give a practical example of the use of the two new sprite calls discussed here, the program in listing 3 executes in a scaled window that can be altered using the mouse. When you run it you will see a small window near the bottom of the screen in which randomly coloured circles are being drawn. By pressing the Select button on the mouse you can move the base position of the display window, while pressing Adjust allows you to alter its size. All this is accomplished without disturbing the drawing process.

To see how this is achieved, take a look at PROCvisuals near the end of the program, and you will see that the circles are apparently being drawn on a full screen. In fact they are drawn into a sprite called "Image", which is then scaled and plotted each time around the drawing loop. To keep the program short, I have just drawn random circles, but PROCvisuals could be replaced by a much longer and more interesting procedure of your choice.

The key to the way in which the program works is the set of three sprite calls (at lines 210, 240 and 270) made within the REPEAT loop. The first sends VDU output to a sprite called "Image" which is the size of a full screen. The second resets VDU output to normal, and the third plots the "Image" sprite suitably scaled. As I hope this example shows, experimentation with these powerful calls will pay dividends.

### Listing 3

```
10 REM >Ros3Sprt
20 REM Program windowing
30 :
40 MODE13:OFF:*POINTER
```

```
50 ON ERROR MODE12:REPORT:PRINT" at 1
ine ";ERL:END
60 DIM sp &15000:!sp=&15000
70 DIM scale 16
80 sp!4=0:sp!8=16:sp!12=16
90 :
100 scale!8=1280:scale!12=1024
110 !scale=320:scale!4=256
120 xbase%=100:ybase%=100
130 :
140 REM Get empty sprite
150 MOVE 0,0:MOVE 1279,1023
160 SYS "OS_SpriteOp",270,sp,"image",1
170 :
180 N%=0
190 REPEAT
200 REM VDU to sprite
210 SYS "OS_SpriteOp",316,sp,"image"
TO R0,R1,R2,R3
220 PROCvisuals
230 REM Reinstate screen
240 SYS "OS_SpriteOp",R0,R1,R2,R3
250 REM Display sprite
260 WAIT
270 SYS "OS_SpriteOp",308,sp,"image",
xbase%,ybase%,0,scale
280 PRINTTAB(10,5)"Frame number ";N%
290 PRINTTAB(8,7)"Mouse:Move,Quit,Scale"
300 N%+=1
310 MOUSE x%,y%,z%
320 IF z%<>0 THEN
330 CASE z% OF
340 WHEN 1:IF x%>xbase% THEN xmult%=
x%-xbase%:!scale=xmult%
350 IF y%>ybase% THEN ymult%=
y%-ybase%:scale!4=ymult%
360 WHEN 4:xbase%=x%:ybase%=y%
370 ENDCASE
380 WAIT:CLS
390 ENDIF
400 UNTIL z%=2
410 ON:END
420 :
430 DEFPROCvisuals
440 GCOL RND(63) TINT 64*RND(3)
450 CIRCLE FILL RND(1100),RND(1000),R
ND(300)
460 PRINT TAB(6,12)"RANDOM SPHERES IN
MODE 13"
470 ENDPROC
```

# RiscBASIC

## ARCHIMEDES BASIC V COMPILER THE BENCHMARKS

† BENCHMARK Name	† BASIC V secs	† ABC secs	† RiscBASIC secs	RiscBASIC /ABC Ratio	RiscBASIC /BASIC V
Repeat..Until(100000)	11.15	1.61	0.08	20.1 : 1	139.4 : 1
While..Loop (100000)	8.70	1.58	0.11	14.4 : 1	79.1 : 1
For..Next (1000000)	20.64	15.98	1.20	13.3 : 1	17.2 : 1
String_Array(10000)	0.99	1.24	0.17	7.3 : 1	5.8 : 1
Integer_Array (10000)	1.45	0.47	0.07	6.7 : 1	20.7 : 1
Real_Array (10000)	1.54	1.52	0.69	2.2 : 1	2.2* : 1
Sieve (1651 Primes)	5.19	0.52	0.07	7.4 : 1	74.1 : 1
Fibonacci	8.17	1.30	0.14	9.3 : 1	58.4 : 1
Ackerman	4.53	0.27	0.17	1.6 : 1	26.6 : 1
Grafscrn	1.67	0.95	0.80	1.2 : 1	2.1 : 1
Textscrn	2.51	2.29	2.24	1.02 : 1	1.1 : 1
Realmath	0.25	0.31	0.26	1.2 : 1	0.96* : 1
Triglog	1.20	3.42	3.42	1.0 : 1	0.35* : 1
Intmath	1.76	0.37	0.17	2.2 : 1	10.4 : 1

\*Using Floating Point Emulator † All benchmarks have the ESC key enabled for fair comparison

### BASIC V Syntax & Keywords

	ABC	RiscBASIC
Full Array Manipulation Operations	N	Y
Local Variables with true scope support	N	Y
Multiple entry to FOR, REPEAT...	N	Y
Unlimited array sizes & dimensions	N	Y
Runtime error handling & reporting	N	Y
Full syntax implementation	N	Y
SUM	N	Y
COUNT	N	Y
WIDTH	N	Y
EVAL, INSTALL, LIBRARY	N	N

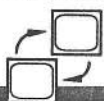


RiscBASIC is the first true BASIC V syntax compiler which produces compiled code that executes up to 20 times faster than the rival ABC compiler and up to 139 times faster than interpreted BASIC V. The Syntax comparison table also demonstrates the completeness of RiscBASIC in implementing all compilable features in the interpreted BASIC V standard with only uncompileable features remaining unsupported.

Additional features include an optimising compiler, register variables for speed, relocatable modules, full cross references, Window-based or command line compilation environment, standalone code generation, in-line assembler with powerful floating point mnemonic extensions, Arthur & RiscOS support, comprehensive compiler directives, plus free updates for future extensions from a leading Software House renowned for technical excellence that makes RiscBASIC the best BASIC V compiler money can buy.

**RiscBASIC : £99.95 Inc VAT & Carriage**

Available from your local dealer or direct by Cheque/Access from  
**SILICON VISION LTD, SIGNAL HOUSE, LYON ROAD, HARROW, MIDDLESEX,**  
**HA1 2AG, Tel: 01-422 2274, 01-861 2173, Fax: 01-427 5169, Tlx: 918266.**



# UNIVERSAL SCREEN MODE CONVERTER

David Spencer shows the easy way to convert between modes.

This program requires RISC OS.

The simple Basic procedure given here will convert a screen image from any mode (other than modes 3, 6 or 7), to any other mode. It is primarily designed to convert to a mode superior to the current one, that is one with at least as high a resolution, and at least as many colours. It will however perform any conversion. There are numerous uses for such a conversion. For example, you might want to edit, say, a mode 13 screen in an art package which only supports mode 15. Or, you may need to convert a screen to a different mode before it can be printed using a particular dump program.

The listing given here contains both the conversion procedure, and also a short sample program to demonstrate it. This will load a saved screen of any mode, and convert it to mode 15. You must have a saved screen with the filename 'Screen' available before running this.

To use the routine in your own code, you merely need to include the definition of *PROCconvert*, and with the screen to be converted displayed, execute:

```
PROCconvert(mode)
```

where *mode* is the resulting mode. The conversion takes just a couple of seconds.

Each time the procedure is called it will claim a block of equal to the size of the source screen mode. This will not normally cause problems because it is likely that the conversion will be performed once only. However, it is possible to move the DIM statements to outside the procedure, and dimension the block *screen* to the size of the largest screen size that will be converted. The procedure can then be called without it reclaiming memory. *PROCconvert* also requires the RISC OS *Colour Trans* module to be installed. If this is not present, the procedure attempts to load it from the *!System* application on the RISC OS *Applications* disc (where it normally resides). This will only work if the *!System* application has been 'seen' in a Desktop directory viewer since the last hard reset. You may wish to make a copy of the module onto your working disc, and change the pathname at the end of line 1010 (after the *RMLOAD*) to reflect this.

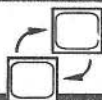
## COLOURS

The way in which colours are translated depends on whether the screen is being converted to a mode with more or less colours than the current mode. In the case where the conversion is to a mode with less colours, each colour from the current palette will be mapped to the closest colour in the new mode's palette. No attempt is made to redefine the palette. This means that the colours used in a picture may be changed considerably. On the other hand, if the new mode offers more colours (or the same number) then its palette is set to the same values as the source mode. Therefore, the colours will be retained exactly. The only exception to this is when the new mode has 256 colours. In this case the palette is not changed, although the large number of available colours mean that the conversion can be performed with no noticeable change of colour.

## HOW IT WORKS

The procedure starts by grabbing the source screen into a user sprite area. It then calculates the size of the source and destination screens, and uses the *Colour Trans* module to perform the palette mapping. The screen mode is then changed and the sprite replotted using the appropriate scaling and colour translation.

```
10 REM >Convert
20 REM Program      Mode Converter
30 REM Version      A1.00
40 REM Author       David Spencer
50 REM RISC User    May 1989
60 REM Program      Subject to Copyright
70 :
80 *SCREENLOAD "Screen"
90 PRINT TAB(0,0);MODE
100 PROCconvert(15)
110 PRINT TAB(0,1);MODE
120 END
130 :
1000 DEF PROCconvert(dmode)
1010 *RMENSURE ColourTrans 0 RMLOAD <System$Path>Modules.Colours
1020 SYS "OS_Byte",135 TO ,,smode
1030 SYS "OS_ReadModeVariable",smode,3
TO ,,scols
1040 SYS "OS_ReadModeVariable",smode,4
```



# UNIVERSAL SCREEN MODE CONVERTER

```
TO ,,sxeig
1050 SYS "OS_ReadModeVariable",smode,5
TO ,,syieg
1060 SYS "OS_ReadModeVariable",smode,7
TO ,,size
1070 SYS "OS_ReadModeVariable",smode,11
TO ,,sx
1080 SYS "OS_ReadModeVariable",smode,12
TO ,,sy
1090 SYS "OS_ReadModeVariable",dmode,3
TO ,,dcols
1100 SYS "OS_ReadModeVariable",dmode,4
TO ,,dxeig
1110 SYS "OS_ReadModeVariable",dmode,5
TO ,,dyeig
1120 SYS "OS_ReadModeVariable",dmode,11
TO ,,dx
1130 SYS "OS_ReadModeVariable",dmode,12
TO ,,dy
1140 DIM map 256, screen size+64, scale
15, pal 64
1150 !scale=dx:scale!4=dy:scale!8=sx:scale!12=sy
1160 !screen=size+64:screen!4=0
1170 screen!8=16:screen!12=16
1180 SYS "OS_SpriteOp",&110,screen,"sprite",0,0,0,sx<<sxeig,sy<<syieg
1190 IF dcols>=scols AND dcols<63 THEN
1200 dp=pal
1210 FOR loop=0 TO scols
1220 SYS "OS_ReadPalette",loop,16 TO ,,col
1230 pal!(loop*4)=col
1240 NEXT
1250 ELSE
1260 dp=0
1270 ENDIF
1280 SYS "ColourTrans_InvalidateCache"
1290 SYS "ColourTrans_SelectTable",-1,-1,dmode,dp,map
1300 MODE dmode
1310 IF dp THEN
1320 FOR loop=0 TO scols
1330 col=pal!(loop*4)
1340 VDU 19,loop,16,col>>8,col>>16,col>>24
1350 NEXT
1360 ENDIF
1370 SYS "OS_SpriteOp",&134,screen,"sprite",0,0,0,scale,map
1380 SYS "OS_SpriteOp",&109,screen
1390 ENDPROC
```

**RU**

## A PARTIAL RENUMBER UTILITY (continued from page 11)

```
1630 .cloop CMP R2,#1:BEQ prtnum2
1640 SWI &120:SUB R2,R2,#1:B cloop
1650 .prtnum2 SWI "OS_Write0"
1660 LDMFD R13!,{R0-R3}:MOV PC,R14
1670 :
1680 .candg MOV R0,#&8F00
1690 ADD R0,R0,#1:MOV R1,#0
1700 .candg2 LDRB R2,[R0,#1]
1710 LDRB R3,[R0]:ADD R2,R2,R3,LSL #8
1720 CMP R3,#&FF:ADDNE R1,R1,#1
1730 LDRNEB R3,[R0,#2]:ADDNE R0,R0,R3
1740 BNE candg2:ADD R0,R0,#4
1750 BIC R4,R0,#3:MOV R3,R1,LSL #2
1760 SWI "OS_GetEnv":ADD R0,R3,R4
1770 MOV R5,R4:CMP R0,R1:ADRHI R0,nrerr
1780 LDMHIFD R13!,{R14}
1790 ORRHIS PC,R14,#1<<28
1800 MOV R0,#&8F00:ADD R0,R0,#1
1810 .candg3 LDRB R2,[R0,#1]
1820 LDRB R3,[R0]:ADD R2,R2,R3,LSL #8
1830 STR R2,[R4],#4:CMP R3,#&FF
1840 LDRNEB R2,[R0,#2]:ADDNE R0,R0,R2
1850 BNE candg3:MOV PC,R14
1860 :
1870 .getrange MOV R11,R14:MOV R3,#0
1880 LDR R0,[R12,#12]:CMP R0,#0
1890 BEQ nostart:BL gnum:MOV R3,R0
1900 .nostart LDR R0,[R12,#16]
1910 MOV R4,#&FF00:SUB R4,R4,#1
1920 CMP R0,#0:BEQ noend
1930 BL gnum:MOV R4,R0
1940 .noend MVN R5,#0:MOV R6,#0
1950 MOV R2,#&8F00:ADD R2,R2,#1
1960 .gr LDRB R0,[R2,#1]
1970 LDRB R1,[R2]:ADD R0,R0,R1,LSL #8
1980 CMP R0,R3:BCC nolo:CMP R6,#0
1990 MOVEQ R7,R2:MOVNEQ R6,#0
2000 .nolo CMP R0,R4:LDRLSB R8,[R2,#2]
2010 ADDLS R8,R2,R8:CMP R6,#0
2020 MOVEQ R5,R0:CMP R0,#&FF00
2030 LDRCCB R0,[R2,#2]:ADDCC R2,R2,R0
2040 BCC gr:CMP R8,R7:ADRLS R0,bserr
2050 LDMLSFD R13!,[R14]
2060 ORRLSS PC,R14,#1<<28:MOV PC,R11
2070 ]NEXT
2080 SYS "OS_File",10,"Renumber",&FFC,,code,P%
```

**RU**



# ARC PROCEDURE LIBRARY

by Lee Calcraft

**This month our procedures automate the process of drawing selection boxes, and of determining which, if any, has been selected by the user.**

The procedures which we introduced last month permitted comprehensive polling of the mouse, and checking for the position of the pointer. This month we will complement these with a procedure for drawing sets of coloured selection boxes, each of which may contain text of the user's choice, and a function which will return the number of the currently selected box.

To avoid duplicating information in the procedure which draws the boxes, and the function which polls them, I have used an array to hold all common data. This means that the system is very easy to use. You simply set up one array for each set of selection boxes, then call PROCdrawgrid to draw it, and use FNgetboxno to check which box in the current set the user has chosen.

0	X co-ordinate of base
1	Y co-ordinate of base
2	Cell width
3	Cell height
4	No of cells across
5	No of cells down
6	Total width (calculated)
7	Total height (calculated)
8	Selection background colour
9	Grid line colour
10	Text colour
11	Text X offset
12	Text Y offset

**Table 1. The Elements in a Selection Box Array**

## THE ARRAY

Table 1 gives the use to which each of the 13 array elements is put. Each must be set up at the start of the program (as in PROCinitialise in the example), except for elements 6 and 7, which are calculated within the program when the grid is first drawn. The function of each should be more or less self explanatory. The first two are the co-ordinates of the bottom left-hand corner of the selection area. If the first of these, the X co-ordinate, is given as -1, then the selection grid will automatically be centred horizontally (see the lower grid in the example

program). The next two co-ordinates are the width and height of a single selection cell. Then comes the number of cells in the X and Y directions, followed by the colour numbers for the background, the selection grid itself, and the text. Finally there are two offsets for the text. These should be adjusted by trial and error to ensure that the text falls symmetrically within a given set of boxes.

## PROCdrawgrid

The grid drawing procedure has just two parameters: the name of the array holding the data, and a keyword used to identify the block of text to be used with the grid. If no text is required, then this should be given as a double quote, as with the second grid in the example.

In order to avoid using line number references (which in any case are not permitted when using INSTALLED libraries), I have made use of a special procedure PROCrestore. This will restore any block of data, located by the label passed as a parameter. In the accompanying example, the word "Fruits" identifies the group of four fruits whose names appear in the first menu grid.

## FNgetboxno

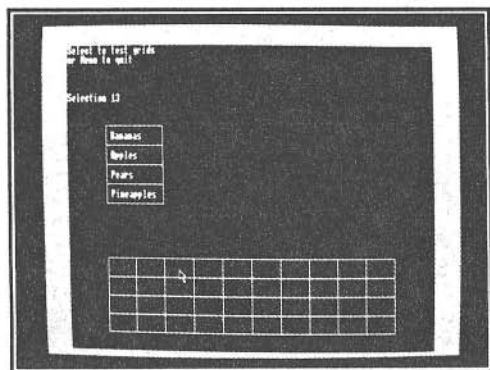
This function takes three parameters: the X and Y co-ordinates of the pointer at the time of selection, and the name of the array corresponding to the grid that you wish to check. It returns the number of the box selected, or -1 if the pointer is outside the designated area.

To see how the procedure-function pair are used together, take a look at the example. This contains all the code you need, except for the definitions of PROCmousewait and FNpositioned from last month. These should both be appended to this month's program.

When you run the program you will see two different selection grids, one with text, the other without. If you press the Select button, a number will be displayed giving the number of the box under the pointer (or zero if you are not over one of the two grids). Note the way in which the pro-



gram increments by ten the grid number returned by the second grid. This results in a unique number being returned for each one of the 44 possible choices, and would allow you to use a single CASE statement to deal with any selection made anywhere on the screen.



Two sets of selection boxes in use

Because of the use of arrays to hold the data for each selection grid, it is a very simple matter to change any feature of any of the grids. For example, to make the first grid two columns wide, alter `area1%(4)` from 1 to 2 in `PROCinitialise`, and add four more words to the data at the end of the program. This is *all* that you need to do. The selection function does not need to be altered at all. It is even easier to change the position of a box on screen, and almost as easy to add a completely new set of boxes by introducing a third array.

*NEXT MONTH we will add a touch of class to our selection screens, and provide a procedure for creating three-dimensional selection boxes.*

```

10 REM >Mse2prcs4d
20 REM Procedure/Function Library
30 REM Version A 0.4
40 REM Author Lee Calcraft
50 REM RISC User May 1989
60 REM Program Subject to Copyright
70 :
80 MODE12:VDU19,0,24,0,192,128
90 ON ERROR REPORT:PRINT" at line ";E
PROC:ON:END
100 PROCinitialise
110 PRINT"Select to test grids"
```

```

120 PRINT"or Menu to quit"
130 PROCdrawgrid(area1%(),"Fruits")
140 PROCdrawgrid(area2%(),"")
150 OFF:*POINTER
160 REPEAT
170 PROCmousewait(-8)
180 IF z%=4 THEN
190 CASE TRUE OF
200 WHEN FNgetboxno(x%,y%,area1%()
)>0:choice%=FNgetboxno(x%,y%,area1%()
210 WHEN FNgetboxno(x%,y%,area2%()
)>0:choice%=10+FNgetboxno(x%,y%,area2%()
)
220 OTHERWISE choice%=0
230 ENDCASE
240 PRINTTAB(0,5);"Selection ";choic
e%," "
250 ENDIF
260 UNTIL z%=2
270 ON
280 END
290 =====
300 DEFPROCinitialise
310 DIM area1%(12)
320 DIM area2%(12)
330 REM.....First group
340 area1%(0)=140 :REM X coord
350 area1%(1)=500 :REM Y coord
360 area1%(2)=200 :REM Cell width
370 area1%(3)=64 :REM Cell height
380 area1%(4)=1 :REM Cells across
390 area1%(5)=4 :REM Cells down
400 area1%(8)=4 :REM Box bcg blue
410 area1%(9)=7 :REM Box line white
420 area1%(10)=7 :REM White text
430 area1%(11)=16 :REM Text x offset
440 area1%(12)=16 :REM Text y offset
450 REM.....Second group
460 area2%(0)=-1 :REM Neg so centre
470 area2%(1)=64
480 area2%(2)=100
490 area2%(3)=64
500 area2%(4)=10
510 area2%(5)=4
520 area2%(8)=1 :REM Red bcg
530 area2%(9)=3 :REM Yellow lines
540 area2%(10)=0 :REM Black text
550 area2%(11)=4
560 area2%(12)=4
570 ENDPROC
580 =====
590 DEFFNgetboxno(x%,y%,a%())
600 REM Returns no of selected box
610 REM Counting from top left (=no 1)
```



```

620 REM and moving left to right
630 LOCAL ix%,iy%,no%
640 IF FNpositiond(x%,y%,a%(0),a%(1),a
%(6),a%(7)) THEN
650   ix%=(x%-a%(0)) DIV a%(2)+1
660   iy%=a%(5)-(y%-a%(1)) DIV a%(3)-1
670   no%=ix%+a%(4)*iy%
680 ELSE no%=-1
690 ENDIF
700 =no%
710 =====
720 DEFPROCdrawgrid(a%(),text$)
730 REM Draws simple selection grid
740 LOCAL ix%,iy%,no%
750 a%(6)=a%(2)*a%(4)
760 a%(7)=a%(3)*a%(5)
770 IF a%(0)=-1 THEN a%(0)=(1280-a%(6)
)DIV 2
780 GCOL a%(8)
790 RECTANGLE FILL a%(0),a%(1),a%(6),a
%(7)
800 GCOL a%(9)
810 FOR iy%=a%(1) TO a%(1)+a%(7) STEP
a%(3)
820   LINE a%(0),iy%,a%(0)+a%(6),iy%
830 NEXT

840 FOR ix%=a%(0) TO a%(0)+a%(6) STEP
a%(2)
850   LINE ix%,a%(1),ix%,a%(1)+a%(7)
860 NEXT
870 IF text$<>" THEN
880   PROCrestore(text$)
890   VDU5:GCOL a%(10)
900   FOR iy%=a%(1)+a%(7) TO a%(1)+a%(
3) STEP -a%(3)
910   FOR ix%=a%(0) TO a%(0)+a%(6)-a%(
2) STEP a%(2)
920     MOVE ix%+a%(11),iy%-4-a%(12)
930     READ text$:PRINTtext$
940     NEXT:NEXT
950     VDU4:OFF
960   ENDIF
970   ENDPROC
980   =====
990   DEFPROCrestore(label$)
1000  RESTORE
1010  REPEAT:READ a$:UNTIL a$=label$
1020  ENDPROC
1030  =====
1040  DATA Fruits
1050  DATA Bananas, Apples, Pears, Pinea
ples

```

RU

## "FIRST"

### ADVANCED STATISTICAL SYSTEM

For Acorn Archimedes; BBC Models B, B+ and Master; "Industry Standard" machines

**A powerful and practical tool for Research, Industry, Business, and Teaching**

Integrated, interactive, robust, fast, accurate, modular. Designed to facilitate data critique.

Many data entry options. Full data management and formatted display. Extensive data transforms.

Handles missing variables and data subsets. Scatterplots, regression plots, function plots.

Univariate statistics. Paired and unpaired t tests. Chi-square tests. Nonparametric methods.

Unrivalled REGRESSION facilities eg weighted, through the origin, standardised. Full statistical information.

Residuals, fitted and predicted values. Much more. Correlations. ANOVA. Contour plots of regressions.

Regression DIAGNOSTICS - VIFs, influential points etc. Automatic warnings.

ROBUST regression - many influence functions. Powerful NONLINEAR least squares

Time Series, eg moving averages, exponential smoothing, causal models.

One- Two- and Three-way ANOVA subsystems. Distributions generator. Histograms. Linear Calibration.

Cluster analysis. Many other powerful analytical and descriptive features. Full utilities.

**Price £120-£180 (machine and version dependent). Special RISC USER discount available.**

There's far too much to describe here. Get full information now from:

**Serious Statistical Software, Lynwood, Benty Heath Lane, Willaston, South Wirral L64 1SD Tel. 051 327 4268**

# **EXPANSION CARDS FOR THE ACORN ARCHIMEDES COMPUTER SYSTEM**

**IEEE488 INTERFACE** a full implementation of the standard for automatic test and measurement systems

**16 BIT PARALLEL I/O** two 16 bit input or output ports with handshake lines for digital control applications

**DUAL RS423 SERIAL INTERFACE** for communicating with two additional RS423 or RS232 devices eg printers, plotters, instruments, etc

**12 BIT ANALOGUE I/O** in development

All the above high performance expansion cards are supplied with high level software for ease of use and a comprehensive user guide.

Take advantage of Intelligent Interfaces' expertise and purchase a complete Archimedes Computer System.

Officially appointed Acorn Scientific Dealer.



**Intelligent Interfaces Ltd**

43b Wood Street  
Stratford-upon-Avon  
Warwickshire  
CV37.6JQ

Tel: 0789 415875

Telex: 312242 MIDTLX G



# Archimedes Visuals

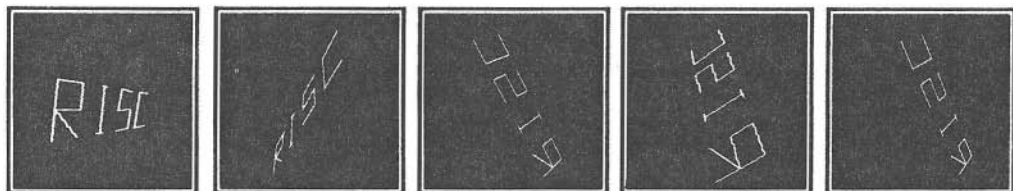
## Spin text around the screen with this program by James Bonfield

This month's Visuals contribution is a program which will spin a piece of text on screen about a central point. The text is generated by the program itself using its own data, and the spinning effect uses 3D drawing techniques which include the use of perspective. Even though the letters are drawn relatively crudely (in order to keep the data statements to a minimum), the spinning effect works well.

When you run the program it asks for a message. This may be up to ten characters in

length (though best effects are obtained with less), and these must be in upper case. After a short pause, the message will appear, and will spin around a central point, with a continually varying plane of rotation.

If you would like to try out the program without typing in all the data, enter all lines up to 930, and when prompted for text, enter hashes (#) or exclamation marks (!). Additionally, the program may be customised a little by experimenting with the text size (*size%*), perspective (*per%*) and depth (*depth%*) in line 80.



A sequence of images using 'RISC'

```

10 REM >Rotate
20 REM Program Text Rotation
30 REM Version A 0.6
40 REM Author James Bonfield
50 REM RISC User May 1989
60 REM Program Subject to Copyright
70 :
80 size%=15:depth%=50:per%=2
90 MODE 0:OFF
100 ON ERROR MODE 0:REPORT:PRINT" at 1
ine ";ERL:END
110 PROCstartdata
120 PROCdotext
130 REPEAT
140 PROCdisplay
150 UNTIL FALSE
160 :
170 DEFPROCstartdata
180 ORIGIN 640,512
190 theta=PI/50:psi=PI/80:alpha=PI/30
200 SA=SIN(theta):CA=COS(theta)
210 SB=SIN(alpha):CB=COS(alpha)
220 SC=SIN(psi):CC=COS(psi)
230 XX=CB*CC:XY=CC*SA*SB-SC*CA
240 XZ=CA*SB*CC+SA*SC
250 YX=CB*SC:YY=SA*SB*SC+CA*CC
260 YZ=CA*SB*SC-SA*CC
270 ZX=-SB:ZY=SA*CB:ZZ=CA*CB
280 TY=0:TX=0:bank%=1
290 ENDPROC
300 :
310 DEFPROCdotext
320 INPUT "Please enter message : "M$
330 PRINT"Please wait"
340 number_of_points=0
350 FOR I%=1 TO LEN(M$)
360 PROCrestore_data
370 READ S
380 number_of_points+=S
390 NEXT
400 DIM points(number_of_points,4)
410 S1=1
420 FOR I%=1 TO LEN(M$)
430 PROCrestore_data
440 READ S
450 FOR N=1 TO S
460 READ points(S1,1),points(S1,2),p
oints(S1,4)

```

# Archimedes Visuals

```

470  points(S1,1)=points(S1,1)+(I%-LE
N(M$)/2)*10-7
480  points(S1,2)=points(S1,2)-4:S1=S
1+1
490  NEXT
500  NEXT
510  ENDPROC
520  :
530  DEFPROCdisplay
540  A=0:X=points(0,1):Y=points(0,2)
550  Z=points(0,3)
560  PROCperspective
570  OX%=XS%:OY%=YS%
580  SYS 6,112,bank%
590  bank%=bank% EOR 3:SYS 6,113,bank%
600  WAIT:CLS
610  FOR A=1 TO number_of_points
620  X=points(A,1)
630  Y=points(A,2)
640  Z=points(A,3)
650  PROCperspective
660  IF points(A,4)<>0 THEN LINE OX%,O
Y%,XS%,YS%
670  OX%=XS%:OY%=YS%
680  NEXT
690  S--(S<12)/5
700  ENDPROC
710  :
720  DEFPROCperspective
730  XT=X*XX+Y*XY+Z*XZ
740  YT=X*YX+Y*YY+Z*YZ
750  ZT=X*ZX+Y*ZY+Z*ZZ
760  points(A,1)=XT:points(A,2)=YT
770  points(A,3)=ZT
780  XS%=size%*XT*depth%/(ZT+per%+depth
%)
790  YS%=size%*YT*depth%/(ZT+per%+depth
%)
800  ENDPROC
810  :
820  DEFPROCrestore_data
830  RESTORE
840  REPEAT
850  READ a%
860  UNTIL a%=ASC(MID$(M$,I%,1))
870  ENDPROC
880  :
890  DATA 31,1,0,0,0
900  DATA 32,1,0,0,0
910  DATA 33,4,3,0,0,3,1,1,3,4,0,3,8,1
920  DATA 34,4,2,5,0,2,8,1,4,5,0,4,8,1
930  DATA 35,8,2,0,0,2,8,1,4,0,0,4,8,1,
0,3,0,6,3,1,0,5,0,6,5,1
940  DATA 36,8,0,1,0,6,1,1,6,4,1,0,4,1,
0,7,1,6,7,1,3,0,0,3,8,1
950  DATA 37,6,0,0,0,6,8,1,6,1,0,5,2,1,
1,6,0,0,7,1
960  DATA 38,7,6,0,0,0,5,1,2,8,1,4,5,1,
0,2,1,2,0,1,6,3,1
970  DATA 39,4,2,5,0,3,6,1,3,7,1,2,7,1
980  DATA 40,4,3,0,0,2,2,1,2,6,1,3,8,1
990  DATA 41,4,3,0,0,4,2,1,4,6,1,3,8,1
1000 DATA 42,8,0,0,0,6,8,1,6,0,0,0,8,1,
3,0,0,3,8,1,0,4,0,6,4,1
1010 DATA 43,4,3,0,0,3,8,1,0,4,0,6,4,1
1020 DATA 44,4,3,-1,0,4,0,1,4,1,1,3,1,1
1030 DATA 45,2,0,4,0,6,4,1
1040 DATA 46,5,3,0,0,4,0,1,4,1,1,3,1,0,
3,0,1
1050 DATA 47,2,0,0,0,6,8,1
1060 DATA 48,6,0,0,0,6,0,1,6,8,1,0,8,1,
0,0,1,6,8,1
1070 DATA 49,5,2,0,0,4,0,1,3,0,0,3,8,1,
2,6,1
1080 DATA 50,6,6,0,0,0,0,1,6,5,1,6,8,1,
0,8,1,0,5,1
1090 DATA 51,6,0,0,0,6,0,1,6,4,1,0,4,1,
6,8,1,0,8,1
1100 DATA 52,5,0,8,0,0,3,1,6,3,1,3,0,0,
3,6,1
1110 DATA 53,6,0,0,0,6,0,1,6,4,1,0,4,1,
0,8,1,6,8,1
1120 DATA 54,6,2,8,0,0,4,1,0,0,1,6,0,1,
6,4,1,0,4,1
1130 DATA 55,3,0,8,0,6,8,1,2,0,1
1140 DATA 56,7,0,0,0,6,0,1,6,8,1,0,8,1,
0,0,1,0,4,0,6,4,1
1150 DATA 57,5,6,0,0,6,8,1,0,8,1,0,4,1,
6,4,1
1160 DATA 58,10,2,2,0,3,2,1,3,3,1,2,3,1
,2,2,1,2,5,0,3,5,1,3,6,1,2,6,1,2,5,1
1170 DATA 59,9,2,1,0,3,2,1,3,3,1,2,3,1,
2,5,0,3,5,1,3,6,1,2,6,1,2,5,1
1180 DATA 60,3,6,0,0,0,4,1,6,8,1
1190 DATA 61,4,0,3,0,6,3,1,0,5,0,6,5,1
1200 DATA 62,3,0,0,0,6,4,1,0,8,1
1210 DATA 63,7,0,8,0,6,8,1,6,4,1,3,4,1,
3,3,1,3,0,0,3,1,1
1220 DATA 64,10,6,0,0,0,0,1,0,8,1,6,8,1
,6,2,1,4,3,1,4,6,1,2,6,1,2,3,1,4,3,1

```

1230 DATA 65,5,0,0,0,3,8,1,6,0,1,1,3,0,  
5,3,1  
1240 DATA 66,10,0,0,0,0,8,1,3,8,1,5,6,1  
3,4,1,6,2,1,4,0,1,0,0,1,0,4,0,4,1  
1250 DATA 67,4,6,0,0,0,0,1,0,8,1,6,8,1  
1260 DATA 68,6,0,0,0,0,8,1,3,8,1,6,4,1,  
4,0,1,0,0,1  
1270 DATA 69,6,6,0,0,0,0,1,0,8,1,6,8,1,  
0,4,0,5,4,1  
1280 DATA 70,5,0,0,0,0,8,1,6,8,1,0,4,0,  
4,4,1  
1290 DATA 71,6,6,8,0,0,8,1,0,0,1,6,0,1,  
6,4,1,3,4,1  
1300 DATA 72,6,0,0,0,0,8,1,6,0,0,6,8,1,  
0,4,0,6,4,1  
1310 DATA 73,6,2,0,0,4,0,1,3,0,0,3,8,1,  
2,8,0,4,8,1  
1320 DATA 74,5,0,0,0,3,0,1,3,8,1,0,8,0,  
6,8,1  
1330 DATA 75,6,0,0,0,0,8,1,0,4,0,6,0,1,  
2,3,0,6,8,1  
1340 DATA 76,3,0,8,0,0,0,1,6,0,1

1350 DATA 77,5,0,0,0,0,8,1,3,4,1,6,8,1,  
6,0,1  
1360 DATA 78,4,0,0,0,0,8,1,6,0,1,6,8,1  
1370 DATA 79,5,0,0,0,0,8,1,6,8,1,6,0,1,  
0,0,1  
1380 DATA 80,5,0,0,0,0,8,1,6,8,1,6,4,1,  
0,4,1  
1390 DATA 81,7,0,0,0,0,8,1,6,8,1,6,0,1,  
0,0,1,3,3,0,7,-1,1  
1400 DATA 82,6,0,0,0,0,8,1,6,8,1,6,4,1,  
0,4,1,6,0,1  
1410 DATA 83,6,0,0,0,6,0,1,6,4,1,0,4,1,  
0,8,1,6,8,1  
1420 DATA 84,4,3,0,0,3,8,1,0,8,0,6,8,1  
1430 DATA 85,4,0,8,0,0,0,1,6,0,1,6,8,1  
1440 DATA 86,3,0,8,0,3,0,1,6,8,1  
1450 DATA 87,5,0,8,0,1,0,1,3,4,1,5,0,1,  
6,8,1  
1460 DATA 88,4,0,0,0,6,8,1,0,8,0,6,0,1  
1470 DATA 89,5,0,8,0,3,4,1,6,8,1,3,4,0,  
3,0,1  
1480 DATA 90,4,0,8,0,6,8,1,0,0,1,6,0,1

RU

## ARCHIMEDES EXPANSION CARDS FROM WILD VISION

### CHROMA 300 -

**Video Overlay and Genlock System.** Enables Archimedes generated graphics and text to be superimposed in full colour on a video signal from a camera or VCR. Ideal for captioning videos.

### HAWK V10 -

**Image Processing System** designed to match the high speed of the Archimedes. Captures, stores and displays up to 4 video images in real time independently of the host Archimedes.

### AD1208 -

**High Speed Analogue/Digital I/O Card** 12-bit, 8 channel analogue to digital converter, 166000 samples per second and 8-bit "user port" digital I/O.

Full information on these easy to fit cards from:



**Wild Vision**  
6 JESMOND ROAD NEWCASTLE UPON TYNE NE2 4PQ  
TEL: 091 281 8481



## Euclid

The 3-D modelling  
and animation system for  
the Acorn Archimedes.



Latest version includes NEW lighting effects and help menu, plus easier ways of colouring and viewing scenes.

Here's what the reviewers say:

"...even half an hour spent with Euclid will suggest many uses in education, professional design, mathematics, programming, and CAD applications: buy it...and you won't be disappointed."

Micronet

"Children enjoy the immediacy of this program and its ability to capture the imagination of even reluctant learners is quite extraordinary."

The Times Educational Supplement 11 November 1988

**EUCLID - Explore a new dimension!**

Price: £45 (inc VAT and P&P).

**Coming Soon:** Orders now being taken for new  
RISCOS only version—£70.

Available by mail order from:

Ace Computing, 27 Victoria Road, Cambridge CB4 3BW.  
Or from your local dealer.





# MOVIE MAKER STAND-ALONE DISPLAY

by Lee Calcraft

RISC User Volume 2 Issues 1 and 2 featured a Movie Maker package which permits the creation of animated sequences of all kinds. As presented, the program only permits playback from within Movie Maker itself, but as promised, we now give a short program which will allow Movie Maker files to be replayed independently. This means that animated sequences can be included within other pieces of software; and as a bonus, a special feature permits the replayed images to be placed anywhere on a mode 13 screen.

To try it out, type in the program, and ensure that you have a machine code file called DeltaCode (the one used with Movie Maker), and an animation file (as created by Movie Maker) in your current directory. You must then customise the program to suit your requirements. There are four values to adjust in the first few lines of the program. Set *file\$* in line

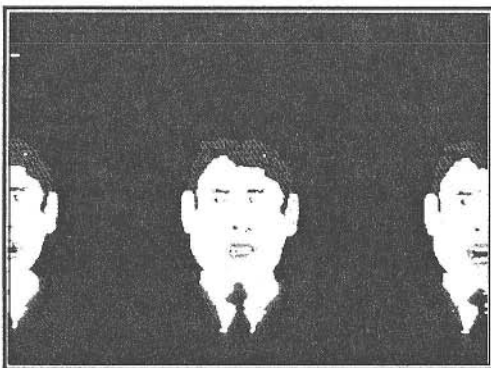
90 to the name of your animation file, and *frames%* to the required playback speed (50, 25, 12, 6 or 3 frames per second). Lines 110 and 120 contain horizontal and vertical off-sets for the image. If these are both zero, the image will appear at the position where it was originally created. The vertical scale runs from 0 to 255, and the horizontal from 0 to 319. Negative numbers may be used (to move left or upwards), but you should not let any part of the image go above the top of the screen, or errors will occur.

The program itself is very simple. It uses just three procedures in sequence: PROCinit, PROCload and PROCreplay. This latter takes as its parameter the number of the last frame to be displayed. In our example we have used the

variable *frametot* to display the full sequence. But you could if you wish be more creative. For example, you could display the sequence from frame 1 to 13 ten times in succession, and then finish off the display by showing the whole sequence. To do this use:

```
FOR count=1 TO 10
  PROCreplay (13)
NEXT
PROCreplay (frametot)
```

Finally, two more points that might be useful



Animated sequences can be placed anywhere on a mode 13 screen.

when incorporating a display into another program. Firstly, the quantity of RAM reserved for the animation file is automatically tailored to the size of the file itself in order to save RAM, and should therefore not need to be altered. And secondly, every time that PROCreplay is called, the graphics screen is cleared to the background colour used when the animation sequence was original-

ly generated. If you wish to keep other things on screen during this process, simply set up a graphics window (after line 80) around your animation area so that only this part gets cleared. As an example VDU24,200;300;500;700; will set up a window whose bottom left and top right co-ordinates are 200,300 and 500,700.

```
10 REM >SoloAnim
20 REM Program Solo Replay
30 REM Version A 0.7
40 REM Author Lee Calcraft
50 REM RISC User May 1989
60 REM Copyright Lee Calcraft
70 :
80 MODE13:OFF
90 file$="Dman2" :REM Customise
```

Continued on page 32

Mike Williams investigates a new and professional looking draughting system for the Archimedes.

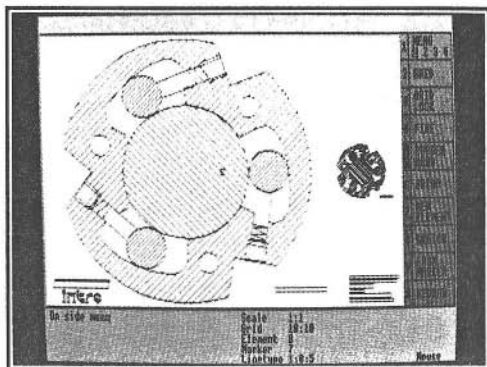
The packaging and documentation of this new product are both very smart leading to a high expectation on the part of the potential user. Within its brief as a 2D draughting system you are unlikely to be disappointed, and in my view the manual is a model of clarity compared with much computer documentation.

Designer Intro is supplied on disc, sealed in an envelope with a strong warning about copyright, and in use the unique serial number of your copy is displayed on screen. The manual, while only some 40 pages in length, is well printed in stiff attractively designed covers.

The separate installation notes cover Arthur 1.2 and hard disc installation, but make no reference to RISC OS. The manual does say that Designer Intro can be selected from the RISC OS Desktop, but my copy did not have the necessary files. However, I found no problems in running Designer Intro under RISC OS by booting the disc as usual. TechSoft says that RISC OS Desktop front-end will be included shortly.

When Designer Intro is first entered, it shows the current default settings for confirmation or change before displaying the drawing screen. These defaults cover drawing size (from A0 to A5), mouse sensitivity (a range of six from 0.1mm to 10mm), and choice of plotter or printer (23 makes including Epson, Graphtec, Hewlett Packard, Plotmate and Watanabe are detailed in the manual).

Once the main drawing area is on view, the foot of the screen shows constantly updated information on the current cursor position and other details. The right-hand edge of the screen is devoted to a ten item menu. However, the first (fixed) item allows the user to cycle through 4 different menus giving 36 menu choices in all, and many of these give further choices, displayed temporarily at the foot of the screen, or prompt for additional user input in the top-most line of the screen display. The result is an uncluttered screen with quick and easy access to a wide range of choices.



Most actions are mouse controlled, with the Select button called *Execute* (and corresponding to 'Y' for response to 'Y/N' type questions), the middle Menu button is designated *Move* (corresponding to 'N'), while the Adjust button, called *Hunt* helps in selecting existing points in a drawing. *Execute* is used to perform the currently selected action, while *Move* locates the cursor at a new position.

Some actions require keyboard input (to specify the size of an angle, for example), while the function keys (F0 to F9) always correspond to menu choices 0-9. Designer Intro can also be switched between mouse and full keyboard control as you wish. I found mouse control much the easier, and this is the default.

All of these basic points are clearly and thoroughly covered at the start of the menu, which then follows with a tutorial section taking you in detail through three separate drawings. These 'walk throughs' are excellent, and do much to convey a good 'feel' for the package. However, I felt that I was then thrown too quickly into the reference section of the manual, and a couple more examples and perhaps some general guidance would be a useful addition at this point.

## BASIC DRAWING

Drawing is delightfully easy. There is a *Rubber-band* feature, as would be expected, or

free-hand drawing can be selected. An *Ortho* option constrains drawing in a horizontal or vertical direction, while a *Grid* may be selected to assist accurate drawing. With the grid selected, *Grid lock* forces the cursor to the nearest grid point, and a *Fine* option, as the name suggests, allows 10 increments between grid points. All of these items are in menu 1.

## CURVES

Menu 2 deals with curve drawing and with transformations of part or all of the current drawing. There are options for circles, ellipses and arcs. The simplest form of circle is defined by moving the cursor to mark its centre and typing its radius in from the keyboard. However, there are also options to draw a circle through a point, touching either one or two lines, arcs or circles, and several more options besides. The same is true for arcs. Thus complex shapes can be readily built up from circles, arcs, tangents and the like, and subsequently deleting the unwanted parts of any construction also seems to work remarkably well (see later), but splines or Bezier curves are not included.

## TRANSFORMATIONS

Other options in menu 2 deal with transformations of your drawing, or selected parts. The possibilities include *Move*, with the option of replicating an object, *Mirror* an object about the x or y axes, and *Rotate* an element through a specified number of degrees. With the latter two options, the transformed drawing can replace the original or be added to the original. There is also a similar *Alter size* option.

In addition to the above, Menu 2 also includes a hatching option. This is like a 'fill' routine except that you determine the angle and spacing of the lines to be used, and cross-hatching is achieved by repeating the process.

Zoom and pan facilities are included in menu 3. There is also a separate scaling option which allows a particular scale factor to be specified. With the default scale of 1:1 and A3 size, the screen drawing area represents 360mm x 260mm. With a scale of 1:2 for example, the displayed drawing size will be twice the default, and so the picture appears smaller.

## DELETIONS

Perhaps the most important option in this menu is *Delete*, and it is worth explaining here a very important feature of Designer Intro. The software creates an internal representation of any drawing seen on the screen, so that at any time the current drawing can be re-created, and more importantly the software can identify any component of a drawing selected by the user. Over and above this, the user can divide a drawing into a series of *Elements*. Thus when rotation is selected, the user specifies which elements of the drawing are to be rotated.

When Delete is selected, a sub-menu allows a choice between 'last item', line or part line, arc/circle or part thereof, element or text. To delete a line or arc you just move the cursor near to that item and press Execute. The software locates that part of the drawing, which then flashes, for you to confirm that deletion should take place. As I said before, deletion seems to work remarkably well, any amount of deletion being handled without problems. The *Redraw* option is useful after multiple deletes may have left 'holes' in various lines. This method of representation and the use of elements also allows any part of an existing drawing to be entered and modified.

The fourth menu allows for text and dimensions (with arrows) to be entered. It also provides for the saving and re-loading of drawings, and a number of other options.

## CONCLUSIONS

The software has a good 'feel' to it in practice, benefiting no doubt from the experience of previous implementations for the BBC micro. Both software and documentation seem excellent, with the one qualification about the manual, and this is a product which can be thoroughly recommended. TechSoft also says that a much enhanced version of this product (called Designer) should be available later this year.

Product	Designer Intro
Supplier	TechSoft UK Ltd, Old School Lane, Erryrys, Mold, Clwyd CH7 4DA. Tel. (082) 43318

# TECHSOFT

## DESIGNER intro

# POSSIBLY THE ONLY DRAUGHTING SYSTEM YOU WILL EVER NEED

COST EFFECTIVE CAD

Created specifically for the Archimedes, **DESIGNER INTRO** is based on its big brother **DESIGNER (ARCHIMEDES)**. It shares the same uniquely logical menu structure and many of the powerful draughting routines. Enough power in fact to cope with all of most peoples draughting requirements. Many of **INTRO's** features are shown below. There is one feature however that we cannot show - its most important one - the simplicity of use that guarantees a short learning curve and speedy drawing generation.

### FEATURES

Fully menu driven with simple prompts and instructions for all routines.

User definable grid with grid lock.

Hunt and lock onto nearest end of line, arc or centre of a arc/circle with a single button press.

True arcs, tangents and normals, essentials for most drawing applications but particularly important for engineering applications. Nine in-built arc types alone.

Ellipses, full or part.

Powerful delete facilities, e.g., last item, line, part line, arc, part arc, etc. (The software automatically finds intersections.)

Unique **MODIFY** command allows drawing to be redrawn a line at a time forwards or backwards. The drawing may be 'entered' at any point thus giving total flexibility to edit and alter drawings.

Automatic or manual dimensioning, user definable arrow heads/leader lines.

Text entry at any size, angle or slope. (Text output limited only by the plotter available.)

Hatching at any angle/spacing. Automatically finds enclosed boundary plus single or multiple islands.

Powerful **ELEMENT** (layer) structure.

No practical limit to the number of drawing elements that may be created.

Comprehensive transformation commands, **MOVE**, **ROTATE**, **MIRROR**, **IMAGE**, **ALTER SIZE**.

No practical limit to the size of drawing created - limited only by disk space.

Libraries of drawings can easily be created. Library items can then be incorporated into the current drawing at any size, angle, position.

Zoom, scale and pan facilities (plot-outs may be taken at any time).

Output to wide range of line plotters from A5 to A0, e.g., Graphtec, Hewlett Packard, Hitachi, Plotmate, Roland, etc. Plus screen dumps for FX/MX/RX compatible printers.

### COST EFFECTIVE CAD FOR YOUR ARCHIMEDES

In creating **INTRO** TechSoft have kept the features that made **DESIGNER (BBC)** so popular. There is however one feature that we have cut back on - the cost. **INTRO** is not a toy (although you will enjoy playing with it), it is a true working system at a realistic price.

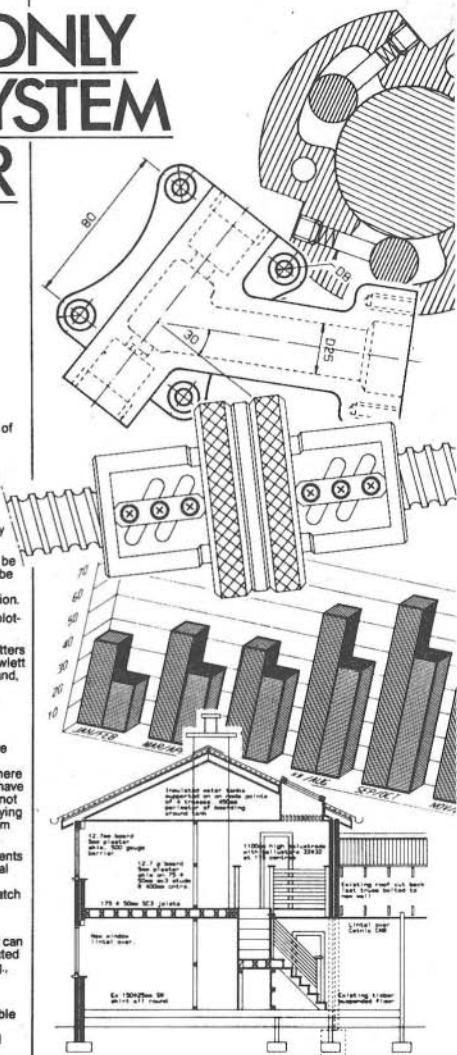
P.S. If your draughting requirements are more demanding than normal take a look at **DESIGNER (ARCHIMEDES)**, more than a match for any drawing.

### PERIPHERALS

Please remember that TechSoft can supply a wide range of CAD related peripherals and accessories, e.g., plotters, plotter pens, etc.

### ORDERING

Cheque with order please, payable to TechSoft UK Ltd.  
Add VAT @ 15% of total. (Official educational orders welcome.)



ONLY **£75** + VAT.

**TECHSOFT**  
NEW IDEAS IN SOFTWARE

TechSoft UK Ltd.  
Old School Lane  
Errys, MOLD  
Chwyd CH7 4DA  
Tel: 082 43 318



# WRITING A MACHINE CODE UTILITY

David Spencer explains the quick way of adding your own star commands.

So-called operating system utilities provide a simple way of adding your own machine code utilities as star commands, without going to the lengths necessary to write a full-blown relocatable module. The Partial Renumber program which appears elsewhere in this issue is a utility of this type, as was the Basic program lister from last month's RISC User.

## CREATING A UTILITY

The operating system recognises a utility command by its filetype, which must be set to &FFC. It is a simple matter to save the utility code, date stamp it, and set the filetype all in one go. This is done using a command such as:

`SYS "OS File",10,"Prog",&FFC,,start,end`  
where *Prog* is the filename, and *start* and *end* are the start and end addresses of the assembled utility. The utility can then be loaded and executed by typing:

`*Prog`

which will cause the operating system to load it into the relocatable module area (RMA), and run it from there. Therefore, all utility code must be fully relocatable.

## PARAMETERS

When a utility is started, register R1 points to any parameters given after the utility name. This will simply be the tail of the star command. This can be seen in the example program given later.

## WORKSPACE

It is highly likely that any utility will require some workspace other than the processor's registers. When a utility is executed, the operating system automatically allocates 1024 bytes of workspace in the RMA. On entry to the routine, register R12 points to the first byte of this workspace, and R13 points to the first byte after the workspace (in other words the address of the last byte plus one). The workspace will always be word-aligned.

It is conventional (but not mandatory) to use register R13 as a stack pointer. This means that the stack, which must be of the full-descending type, will grow down into the 1K workspace, and care must be taken not to overwrite it when using the workspace. In an average program

which uses the stack for storing return addresses and temporary registers, about 100 bytes should be allowed for the stack.

If more workspace is required, this can be claimed using the call SWI "OS\_Module", with a *Claim Block* reason code, as described on page 361 of the *Programmer's Reference Manual*. Any memory claimed in this way must be released before the utility exits. This is also done with SWI "OS\_Module", and is explained on page 362 of the PRM.

## ERRORS

An important point with utilities, and one which could cause many problems, is the handling of errors. Basically, a utility must not directly cause any errors to be generated. Instead, if the utility wishes to generate an error, it must set register R0 to point to an *error block*, and return normally, but with the overflow flag set.

An error block consists of a word-aligned word containing the error number, followed by the text of the error message terminated by a byte containing zero. This is explained on page 9 of the PRM.

The best way of setting the overflow flag before exit is to use the instruction:

`ORRS PC,R14,#1<<28`

This assumes that the return address is in R14. If it has been stacked, then it must be popped again before the above instruction is executed. It is also important to ensure that in the case of a normal exit, the overflow flag is clear. On entry to the utility, this will always be true, so the best method is to restore the flags as well as the return address when exiting. This can be done with:

`MOVS PC,R14`

if the link register has not been stacked, or:

`LDMFD R13!,{PC}^`

if it is stacked.

A further twist is in calling SWI routines, as these also must not be allowed to generate errors. To ensure this, all names must be prefixed with an 'X', for example:

`SWI "XOS Module"`

This means that if an error occurs, rather than



generating it, the SWI will return with the overflow flag set, and register R0 pointing to an error block. If a utility makes a SWI call and finds the overflow flag set on return, it must exit immediately leaving the flag set.

## EXAMPLE

The program given in the listing creates a simple utility which illustrates many of the features discussed. The utility will read a numeric parameter from the command line, and provided the value is non-zero, double it and print the result. In the case of a zero value, an error is given. The utility is saved as 'DEMO', and can be invoked using, for example:

```
*DEMO 123
```

which will print the result 246.

```
10 REM > DemoSrc
20 REM Program      Utility Demo
30 REM Version      A 1.00
40 REM Author       David Spencer
50 REM RISC User    May 1989
60 REM Program      Subject to Copyright
70 :
80 DIM code 100
```

```
90 FOR pass=0 TO 3 STEP 3
100 P%=code
110 [OPT pass
120 \ Read value - R1 points to tail
130 MOV R0,#10:MOV R2,#1<<31
140 SWI "XOS_ReadUnsigned"
150 MOVVS PC,R14
160 CMP R2,#0:BNE notzero
170 ADR R0,errorblock
180 ORRS PC,R14,#1<<28
190 \ Double number into R0 & convert
200 .notzero MOV R0,R2,LSL #1
210 MOV R1,R12:MOV R2,#256
220 SWI "XOS_ConvertCardinal4"
230 MOVVS PC,R14
240 \ Output result + Newline
250 SWI "XOS_Write0":MOVVS PC,R14
260 SWI "XOS_NewLine":MOVVS PC,R14
270 MOVS PC,R14 \ Retain entry flags
280 .errorblock EQU 1234
290 EQU$ "A zero value was given"
300 EQU$ 0
310 JNEXT
320 SYS "OS_File",10,"Demo",&FFC,,code
,P%
```

RU

## Movie Maker Stand-Alone Display (continued from page 27)

```
100 frames%=25      :REM 25 frames/sec
110 xoffset=0        :REM Horiz offset
120 yoffset=0        :REM Vert offset
130 PROCinit
140 PROCload(file$)
150 PROCreplay(frametot)
160 ON
170 END
180 :
190 DEFPROCinit
200 ON ERROR REPORT:PRINT" at line ";E
RL:ON:END
210 handle%=OPENIN(file$)
220 bsize%=EXT# handle%+6400
230 CLOSE# handle%
240 DIM screens bsize%
250 DIM buff $30,code $200
260 OSCLI("LOAD DeltaCode "+STR$-code)
270 speed%=50 DIV frames%
280 A%=screens+bsize%-610
290 CALL code :REM initialise
300 basaddr!=(code+244)
310 !(code+244)=basaddr+xoffset+yoffset
t*320
320 ENDPROC
330 :
340 DEFPROCreplay(stopframe)
350 GCOL 128+startcol TINT starttint
360 CLG:B%=screens+12:N%=0
370 REPEAT
380 A%=B%:N%+=1:B%=USR(code+12)
390 IF speed%>0 THEN
400 FOR S%=1 TO speed%
410 WAIT
420 NEXT
430 ENDIF
440 UNTIL N%=stopframe OR B%=0
450 IF B%>0 THEN A%=B%:framen=N%+1 EL
SE framen=N%
460 ENDPROC
470 :
480 DEFPROCload(file$)
490 OSCLI("LOAD "+file$+" "+STR$-scre
ns)
500 frametot!=screens
510 startcol=screens?4
520 starttint=screens?5
530 ENDPROC
```

RU

# Attention RISCOS Users!!

## A New Communications Package from BBC Soft

Taking full advantage of RISCOS, ArcComm is an exciting, brand-new communications package for the Archimedes that you can't afford to be without!

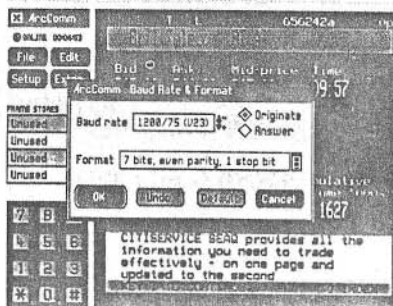
ArcComm's comprehensive design allows access to all three European videotex standards (Prestel, Télétel, and Bildschirmtext), together with VT52, VT102, and ANSI terminals.

It's wonderfully simple to use! All screen displays are clear and uncluttered, with pop-up menus and dialogue boxes, making control as easy as possible.

ArcComm supports Hayes, DTI, and manual modems, with details of how to write drivers for other modems in the User Guide provided. You will need RISCOS, and at least 1Mbyte of memory to use ArcComm.

To order ArcComm, simply write to the address shown, enclosing a cheque. ArcComm is £29.95 inc VAT, plus £1.50 p&p per order.

BBC Soft,  
PO Box 234,  
WETHERBY,  
West Yorkshire,  
LS23 7EU.



# RISC OS BASIC

Mike Williams describes some of the new features to be found in Basic V under RISC OS.

In this article I shall deal with the additions to Basic itself, but note that there are also a number of changes to the Basic assembler which we hope to deal with in a separate article.

## DYNAMIC LIBRARIES

One of the useful features introduced into the original version of Basic V was the facility to access libraries of functions and procedures from within Basic. A set of such routines could be loaded into memory from a file by using the `INSTALL` or `LIBRARY` commands (see RISC User Volume 1 Issue 2). However, the management of this facility was not as flexible as it might have been.

If more than one library is needed, then each will take up additional memory as it is loaded, resulting in the allocation of quite excessive amounts of memory in some instances.

There is a well established way of overcoming this, and that is in the use of overlays. In the past a program would be divided up into a main program and several additional modules, only one of which needed to be in memory at any one time. Provided sufficient memory for the largest module was allocated from the outset as an overlay area, any module could be loaded dynamically as and when required.

That is basically the idea now implemented in Basic V. It allows any one of a series of named libraries to be loaded dynamically as required into a single overlay area, and it works as follows.

The new `OVERLAY` statement specifies a string array, which must already have been dimensioned, containing the names of the library files that may need to be loaded. It allocates an area of memory large enough for the biggest library, though none is loaded at this stage. Then, when a function or procedure is called, Basic V proceeds as follows.

It first searches the current program, it then searches any `LIBRARYs`, then any `INSTALLED` libraries, and finally the overlay area. If the rou-

tine can still not be found it searches through each library file named in the `OVERLAY` array. Provided the routine is found in one of the `OVERLAY` files, that library will then be loaded into the overlay area and the routine executed. For example, we could write:

```
DIM overlay$(10)
overlay$(1)="$$.Library.Maths1"
overlay$(2)="$$.Library.Maths2"
overlay$(3)="$$.Library.Input"
overlay$(4)="$$.Library.Output"
OVERLAY overlay$()
```

to specify the four overlays given above. Any of the four named overlays will be loaded into the overlay area if it contains a function or procedure which cannot otherwise be found.

Unlike the use of `LIBRARY` and `INSTALL`, an overlay routine cannot reference a function or procedure in another overlay for obvious reasons, as all overlays dynamically share the same area of memory. Any overlays no longer required may have their name replaced by a null string in the overlay array, (""), which will speed up any subsequent searching.

Also, new overlay file names may be added to an existing overlay array (provided none is larger than any originally specified, but the names stored there should not otherwise be changed. A new `OVERLAY` statement will effectively re-initialise the whole overlay facility, including a new allocation of memory for the overlay area.

## ARRAY HANDLING ENHANCEMENTS

Although Basic V already allows for all the elements of an array to be initialised to the same value (string or number as appropriate), there has until now been no simple way to set different values, other than by multiple assignment statements. This has now been changed, and the values may be listed after an equals sign ("=") separated by commas, thus:

```
DIM code(5)
code()=129,132,135,131,133,134
```

Remember that the first element of any array is element zero, and there are therefore six values for the dimensioned array in the example above. For multi-dimensional arrays, the order is that

where the right-most subscript changes the quickest.

Two additional functions have now been provided. **SUMLEN** returns the sum of the lengths of all the strings stored in a string array (and is to some extent the string equivalent of the numerical **SUM** function), while **MOD** returns the modulus of the elements of a numeric array, that is the square root of the sum of the squares of the elements, for example:

```
modA=MOD (A ( ) )
```

or just:

```
modA=MOD A ( )
```

## SAVE AND RESTORE DATA POINTER

Just as the error status can be saved to and retrieved from the stack (using **LOCAL ERROR** and **RESTORE ERROR**), the **DATA** pointer can now be similarly saved and recovered. **LOCAL DATA** saves the current **DATA** pointer to the stack, and **RESTORE DATA** retrieves it again. This is most useful where **DATA** statements are included in function and procedure libraries, where **RESTORE+0** (usually - but see below) is used to set the **DATA** pointer to the first item of data in the library routine. On exit from a procedure or function, the **DATA** pointer is automatically reset from the stack (similar to what happens with the error status), and **RESTORE DATA** is not explicitly needed in these circumstances. **LOCAL DATA** must be the last item to be declared **LOCAL** in a function or procedure definition (apart from **LOCAL ERROR**).

Note, that in general, the statement:

```
RESTORE +n
```

will move the data pointer 'n' numbered lines of Basic from the current line number. The form **RESTORE +0** serves the purpose of setting the data pointer in a library procedure or function (if included near the start) to a point in the routine before any **DATA** statements.

## MISCELLANEOUS ENHANCEMENTS

As a further aid to error trapping, a statement of the form:

```
ERROR EXT <number>,<string>
```

will cause the current error handler to be invoked with the error number and message specified (as is the case when **ERROR** alone is used in the same way), but will now cause an

exit from Basic to the supervisor. This can be useful if a Basic program effectively implements a star command so that when an error occurs an exit can be made back to the star prompt.

If Basic is invoked with the command **\*BASIC**, this can optionally be followed by a qualifier and filename. Thus:

**\*BASIC Myprog**

will run the Basic program **Myprog**, and:

**\*BASIC -chain Myprog**

has exactly the same effect, while

**\*BASIC -load Myprog**

will load but not run the program, and:

**\*BASIC -quit Myprog**

will run the Basic program **Myprog**, and quit to RISC OS on termination. The pseudo-variable **QUIT** may now be used in a Basic program to determine if the qualifier **-quit** was used (**TRUE**) or not (**FALSE**).

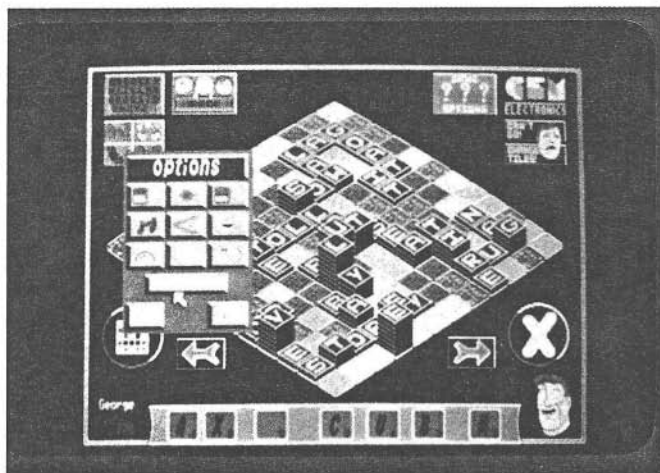
A further change is that the pseudo-variable **END** may now be assigned a value (in a multi-tasking program). In this environment the task manager will allocate an initial amount of memory, but by assigning a suitable value to **END** (during the course of the program), a multi-tasking program can release unwanted memory for use by other tasks (or grab more). Remember that Basic starts at &8000 and will usually require a minimum of about 10K, thus:

```
END=&A000
```

is about the minimum setting for **END** in a multi-tasking context. If a Basic program is not multi-tasking, then this feature should be ignored.

Finally, there are a number of additional or changed error messages:

- |    |  |
|----|--|
| 0  | Incorrect in-core file description             |
| 2  | Assembler limit reached                        |
| 6  | Type mismatch: numeric array needed            |
| 6  | Type mismatch: string array needed             |
| 10 | No room to do matrix multiply                  |
| 11 | No room for this DIM [was error 10]            |
| 11 | No room for this dimension [also 10]           |
| 27 | Missing (                                      |
| 37 | No room for function/procedure call            |
| 42 | DATA pointer not found on stack                |
| 44 | Too many nested structures                     |
| 52 | Can't install library                          |
| 52 | Bad program used as procedure/function library |
| 52 | No room for library                            |



## Word Up Word Down

An innovative 3-dimensional word game.

- \* 3-dimensional and plan view displays
- \* 33,000 + word dictionary
- \* Professionally digitised speech and sound fx
  - \* 1 to 6 players
- \* Computer play options over 3 skill levels
  - \* Rotating 3-dimensional board
- \* Time limit plus other extensive options
- \* User-definable palette and mouse options

**BOTH  
PROGRAMS  
RISC OS  
COMPATIBLE**

### What the magazines have said ...

"... Word Up Word Down, a 3D Scrabble game that really is excellent. It has delightful graphics and it works beautifully mostly under mouse control ... highly recommended"

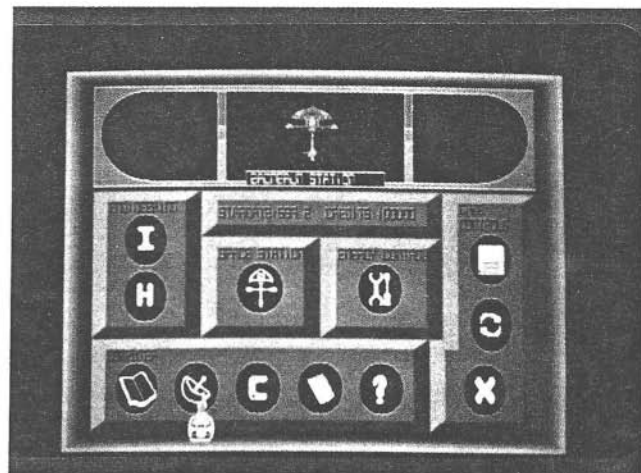
Risc User.

"... Startrader ... absorbing gameplay."

Risc User.

"Outstanding graphics and sound effects."

Archive.



## STARTRADER

A strategic space adventure of epic proportions

- \* 800 unique planets spanning 112 star systems
- \* Professionally digitised sound effects and speech
  - \* 20 different types of spacecraft
- \* Real-time combat and docking sequences
- \* Realistic trading with market fluctuations
  - \* Outstanding graphics
- \* Multi-screen icon based ship control systems
- \* Scientifically based realistic game structure

**£17.95 each**

Prices all inclusive (Overseas orders add £1.00)

Send Cheques/POs to:

**GEM ELECTRONICS, 17, Tandragee Road,  
Portadown, CRAIGAVON, UK. BT62 3BQ.**



# PREMIER TEXT PROCESSING

## Mark Sealey investigates a newcomer to the arena of text and word processing on the Archimedes.

Premier is a difficult piece of software to describe. It does many things that are largely concerned with manipulating text. The package includes a word processor, and the manual was apparently written using this, but the software has much more to offer thanks to what the author of the package calls *dynamic text* and the use of command files.

Most of the tasks that Premier does (and does well) are achieved by the use of an integral command language, and this is at the heart of the whole system. The supplied word processor is an application written using this language, while the user can similarly write his own applications. Using the command language, what is produced - on screen and as hardcopy - can be controlled either in *immediate mode*, or via files of those same commands *compiled* and executed at various points during operation of the software.

### GETTING STARTED

The single disc is accompanied by a 160 page manual. There is also a keystrip for the word processor application, and two blank keystrips for your own applications. When booted or selected from the Desktop (Premier runs under RISC OS, though will not yet multi-task), a default set of values is loaded if you provide no other parameters. These defaults set the screen to mode 12, and redefine all the keys to sensible standard functions.

The standard screen display uses a muted set of colours, and you quickly become accustomed to the various conventions used by Premier. One aspect that needs getting used to is that commands entered through the keyboard are not executed immediately. Instead the backspace key is used to execute a sequence of commands previously entered.

### BUFFERS, MARKERS AND WINDOWS

The tutorial section of the manual takes you through those concepts which are fundamental to Premier, whatever it is doing. In Premier's

environment, there can be any number of *buffers*, which are separate places in memory in which text is kept. *Markers* (unlimited in number but with at least one) identify exact positions in the text and thus within the buffer. Obviously the initial cursor position is one such. Until you name and define a *Window*, no text is visible. Here, for example, a buffer is named and selected and a piece of text is assigned to that buffer. The text is then displayed in a two line window at the bottom right of the screen:

```
buffer "request"
select "request"
display dynamic
insert "Please return your library books by "
insert date
window "footright" = 20,21,20,70
assign "request" "footright"
colour "footright" = blue, ink, white
open "footright"
```

As long as "request" is selected, any text will be entered in the window to which it has been assigned, and be edited with the cursor keys. The date, read from the Archimedes internal clock, will appear thanks to the command *display dynamic*.

### APPLICATIONS

One of the strengths of the package is that whole applications can be written to suit very different requirements. The snippet above might come from an entire Lending Library filing system.

Data files of book titles and borrowers could all be read and manipulated by Premier's hundred or so commands. Letters of varying severity could be sent according to the lateness or persistence of the borrower. Because a variety of arithmetic (as well as logical, boolean and string) operators are available, fines could be calculated, VAT added, and a totally personalised reminder sent. And this is just a start.

### COMMAND FILES

These are likely to be written using the word processor application supplied with Premier,

## ANNOUNCING A MAJOR NEW SOFTWARE RELEASE

# Premier

Circle Software are proud to announce a major new software release for the Archimedes. Premier is so new, there is as yet no defined category for it. Is it a word processor? Is it a report generator? Is it a spreadsheet or a data base? Premier fits none of these categories, rather it spans them all. Premier is a major advance in capability from a single program. These are just some of its features -

The ability to read and execute commands from files allows any text processing to be carried out just by typing a 'star' command, or picking from the desk-top. Further, the ability to define 'macro' commands, and to assign these to ANY key, allows complete 'applications' to be built, including customised spreadsheets, and simple data bases. A command file defining a full featured word processor, with matching key-strip, is included with Premier.

This word processor alone justifies the purchase of Premier. It allows text to be moved between any number of files open at once, searches to be made forwards or backwards using powerful pattern matching, document headers and footers to be defined, has on screen formatting and other features too numerous to list here. The command file defining this may be edited to tailor the features to your own requirements, a feature not found in any other word processor.

DYNAMIC TEXT is a new concept which allows the inclusion of variable items within any document. Such items may show today's date or time, the current value of VAT, or a name and address. The actual text displayed may be computed, defined at the keyboard, or read from a data file. This system facilitates the production of statements and invoices, financial reports, or personalised mail shots.

Premier's powerful command language, which includes maths and string functions, repeat loops and 'if - then' constructs, may be included in the definition of a DYNAMIC TEXT item, allowing full spread-sheet type facilities anywhere on the page, not just in neat rows and columns. This also allows printer codes to be sent, providing the simplest method yet devised to control printer functions from embedded commands.

Now fully RISC-OS Multi-tasking.

Write or phone NOW for full details, of Premier, a £5 refundable demo disc or the full version of Premier.

## Circle Software

33, Restrop View,  
Purton, Swindon,  
Wilts. SN5 9DG  
Tel: 0793 770021



Price including 160 page  
manual and applications,  
£96 + VAT, post free.



and are essentially made up from a superset of the commands already mentioned. Macros (like procedures in Pascal or other structured languages) can be created, as well as multi-line functions. Although plans to develop Premier further include improved arithmetic functions and arrays, there is already a pleasingly rich set of commands - with iteration and conditionals - sufficient to devise quite sophisticated command files.

How would you do this in practice? Assume your library suite has two windows on screen. In the upper you might want the text of the letter which will be sent. In the lower the name and address of the borrower together with a borrowing-record. The command file to do this largely obvious. First define your buffers, and then initialise a variable to represent how late the borrower is:

```
int "overdue" = 0
```

Then calculate the difference between *now* and *then* by reference to the date, assigning the result to the variable *overdue*, and read in the appropriate letter-file:

```
if ( overdue > 1 and overdue <= 2 ) lenient
if ( overdue > 2 and overdue <= 4 ) sharper
if ( overdue > 4 ) severest
```

where *lenient*, *sharper* and *severest* have been defined as macros that will select and load the relevant files from disc into the correct buffer and window.

It should be clear by now that very sophisticated applications can be developed. It has to be said, though, that extensive programming of applications will not necessarily be within the capabilities of all Premier users. It is to be hoped that third parties will eventually write applications, and indeed Circle Software themselves are pursuing this point.

Don't be put off, though. Studying the command file of the word processor application is probably as good a way as any to become familiar with this language, which is not difficult to use. However, the manual is rather poorly indexed, and, as it stands, does not really con-

tain enough detail on this subject. Since it will be the effectiveness of the command files - the programming - which will determine the usefulness of the application, you must bear this in mind when deciding whether to buy Premier.

Having said that, further applications (including a spreadsheet and database) are imminent. It is likely that improvements to the manual will be made to reflect the need for greater clarity, and more detail on the writing of command files. And note that the purchase price includes all such updates and improvements this year.

## FURTHER FEATURES

There is very little not under the user's control. Tabs can be set and colours changed within the palette of each screen-mode. There are constants and some measure of on-screen help for keyword syntax. Pages can be fully formatted and Epson compatible printers addressed directly by control codes; text dialogue can be interactive with an equivalent to Basic's INPUT. There is an impressive array of search functions - with flexible wild-carding, and null as well as literal matching - both forwards and backwards.

## CONCLUSIONS

Clearly, much thought has gone into this highly professional package. The fact that it is both dynamic, open-ended and intended for user-adaptation as well as being under continual development makes it particularly welcome. If you are used to using integrated data suites as in View Professional/Pipedream, or to customising applications to suit your business or hobby, then Premier should definitely be considered. As upgrades appear, it can only grow in appeal.

Product	Premier Text Processor
Supplier	Circle Software 33 Restrop View, Purton, Swindon SN5 9DG.
Price	£110.40 inc. VAT and p&p
Price includes all updates throughout 1989. Demo disc £5 refundable against full purchase.	

# FUNCTION KEY CODES

David Spencer shows how your programs can make the most of the keyboard.

The 102 key keyboard of the Archimedes is very nice, but making use of all the keys from within a program can be a bit tricky. In this article I will show how the system can be programmed to allow all the keys to be read. First, a brief description of how the keyboard is handled.

Each time a key is pressed or released the keyboard sends an *up-down code* to the keyboard handler software within the operating system. The handler then converts this into a *buffer code* which is placed in the keyboard buffer. When the key value is read from the buffer by the program, it is further translated into a code, or series of codes. The following short program will print out the codes of any keys pressed:

```
10 REPEAT key=GET
20 PRINT key " (&";~key;)"
30 UNTIL key=17
```

The buffer codes for the vast majority of keys are exactly the same as the codes returned to the program. These correspond to all the keys which produce printable characters, or standard control codes such as Delete and Return. For example, pressing the 'A' key with the Caps Lock off will enter the value 97 (&61) into the keyboard buffer, and when the program reads this it will read out the value 97 (&61) which is the ASCII code for an 'a'. However, the function keys and cursor keys are very special cases.

## FUNCTION KEYS

Table 1 shows the buffer codes for the function keys (and the cursor keys, but these can be ignored for the moment). From the table, you can see that the code entered into the buffer depends not only on the key pressed, but also the combination of Shift and Ctrl as well. Furthermore, the Print key behaves as if it were function key F0, and the Insert key as if it were F13. When a function key code is removed from the buffer it is not returned directly to the program. Instead, the buffer code is split into two parts - the high and low hex digits (nibbles). For example, the buffer code for key F11 is &CB. This would be split into the high digit &C (12), and the low digit &B (11). The top nibble is then used to look up the

Key	Basic Code	Key +Shift	Key +Ctrl	Key +Shift+Ctrl
Print	&80	&90	&A0	&B0
F1	&81	&91	&A1	&B1
F2	&82	&92	&A2	&B2
F8	&88	&98	&A8	&B8
F9	&89	&99	&A9	&B9
Copy	&8B	&9B	&AB	&BB
<Left>	&8C	&9C	&AC	&BC
<Right>	&8D	&9D	&AD	&BD
<Down>	&8E	&9E	&AE	&BE
<Up>	&8F	&9F	&AF	&BF
Page Down	&9E	&8E	&BE	&AE
Page Up	&9F	&8F	&BF	&AF
F10	&CA	&DA	&EA	&FA
F11	&CB	&DB	&EB	&FB
F12	&CC	&DC	&EC	&FC
Insert	&CD	&DD	&ED	&FD

Table 1. Buffer codes for special keys

action to perform. There are four possible actions:

1. Ignore the keypress.
2. Expand the keypress into a soft key definition.
3. Translate the keypress to a single code.
4. Translate the keypress to a zero value followed by a code.

*FX n	Range covered
221	&C0 - &CF
222	&D0 - &DF
223	&E0 - &EF
224	&F0 - &FF
225	&80 - &8F
226	&90 - &9F
227	&B0 - &BF
288	&B0 - &BF

Table 2

The appropriate calls are listed in table 2.

The action of the keys is set using a command such as:

```
*FX221,n
```

For each of the eight possible buffer code high digits (&8 - &F), there is a separate \*FX call which determines how that buffer code will be interpreted.

## FUNCTION KEY CODES

where  $n$  determines the action for the chosen range of keys. The possible values are:

- 0 Ignore the keypress.
- 1 Expand the key as a soft key.
- 2 Return zero followed by buffer code.
- 3-255 Return  $n + \text{key index}$ .

The first one of these is self-explanatory - the keypress is ignored. The second option causes the keypress to be expanded into a soft key definition (as defined with \*KEY). The individual buffer code determines which key definition is used. For example, \*FX226,1 will select the Shift-Function keys to be expanded into soft keys. Pressing Shift F2 will then enter the buffer code &92 (from table 1). The high digit is 9, and therefore the \*FX226 value will be looked up (table 2). As we have set this to generate the soft key string, the operating system will take the definition corresponding to the low digit (2), and return this.

The third option causes the keypress to be returned as two codes. The first is a zero which indicates 'Function key code follows', and the second is the buffer code of the key. For example, \*FX225,2 will select the function keys (without Shift or Ctrl) to return two-byte codes. If key F8 is now pressed, two codes will be returned - a zero, and an &88. This can be tried out with the program given earlier. The main use of these two byte codes is so that function key presses can be distinguished from foreign characters generated using the Alt key.

The final option allows a single byte code of the form  $n+x$  to be returned, where  $n$  is the value programmed with the \*FX command, and  $x$  is the lower digit of the buffer code. For example, \*FX225,64 will program the function keys to produce codes from 64 upwards. Hence, the Print key will generate ASCII 64 which is '@', F1 will generate 65 ('A'), F2 66 ('B') etc. Obviously the value of  $n$  cannot be 0, 1 or 2, because these values are used to specify the other options.

### DEFAULT VALUES

The default settings for the function keys are shown in table 3. This means that the

function keys on their own will be expanded into the programmed strings, while all the other combinations will produce single byte codes (e.g. &81-&89 for Shift F1 to Shift F9) except for Print and F1-F9 pressed in conjunction with Shift and Ctrl, which will be totally ignored.

Range	Action
&80-&8F	1
&90-&9F	&80
&A0-&AF	&90
&B0-&BF	0
&C0-&CF	1
&D0-&DF	&D0
&E0-&EF	&E0
&F0-&FF	&F0

Table 3. Default settings for function keys

the cursor keys don't cause any value to be entered into the keyboard buffer at all. Instead, they move the input cursor around the screen and Copy 'picks up' the character below the cursor. Their operation can, however, be modified by using the command \*FX4:

- \*FX4 0 Normal operation
- \*FX4 1 Generate fixed codes
- \*FX4 2 Act as function keys

The first of these is the default 'editing' mode described above. The second mode of operation causes the cursor keys to return fixed codes regardless of whether Shift and/or Ctrl are pressed. These codes are:

Copy	135 (&87)
Left	136 (&88)
Right	137 (&89)
Down	138 (&8A)
Up	139 (&8B)

In the third mode, the cursor keys behave just like function keys, adopting the buffer codes listed in table 1.

Finally, the best way to learn how to make best use of the function keys is to experiment with various settings of \*FX221-228 and investigate the codes produced, perhaps using the simple program given earlier.



# ARE YOU GOOD ENOUGH?

As the leaders in software for the Archimedes range of computers, CLARES MICRO SUPPLIES are looking to extend our range even further. We are looking for people who are as excited by the Archimedes as we are.

If you have written any programs, completed or not, then we would like to hear from you.

If you have any ideas for programs and have the ability to execute the ideas then we want to hear from you.

If you have the ability to program the Archimedes but not the ideas to program then we want to hear from you.

Programs can be written in any language as long as they perform their stated task. Many of our programs contain large chunks of BASIC with ARM code in the areas that it is needed. BASIC on the Archimedes is a very powerful language and we do not attach any snob value to its use. If your program does what is meant to do then thats all we are interested in. Why not join the top team on the Archimedes. You get the support of our in-house team, privileged access through us to Acorn and invitations to our informal programmers seminars.

The most important point is that you will be earning top royalty rates of if you prefer we will purchase your program outright.

Please write, in confidence, to Mr. D. Clare at:

**Clares Micro Supplies,  
98 Middlewich Road,  
Northwich,  
CHESHIRE CW9 7DA**

If you have a program either complete or in development then please enclose a copy for our evaluation.

To protect yourself we advise that you lodge a copy of the program with your bank or solicitor BEFORE you send us a copy. You can then prove that your program pre-dates anything that we have.

Act today and become part of the leading software team producing software for the worlds fastest micro.



# WHICH ASSEMBLER?

David Spencer explains the role of an assembler, and compares two such products now available for the Archimedes.

An assembler is a piece of software that takes an assembly language program (the source code), and converts it into the corresponding machine code (the object code) which can then be executed directly by the computer. Basic V on the Archimedes already contains a built-in assembler, and this will probably be adequate for the majority of users writing short to medium length machine code programs. Indeed, the entire application package for Computer Concept's ScanLight (see review last month) was written using Basic's assembler.

However, there is one important case when the Basic assembler is not adequate. This is when it is desired to assemble a machine code program in several sections, and then link them together to form a single program. In fact, some of the sections may actually be compiled high level languages (C or Pascal), rather than hand-coded routines. Additionally, but of less importance, is the Basic assembler's inability to handle the floating point emulator instructions. Both these shortcomings are rectified by the two assembler packages featured here.

## ASSEMBLER FEATURES

Before looking at available assemblers, I will describe briefly some of the features you would expect to find in any Archimedes stand-alone assembler.

### CONDITIONAL ASSEMBLY

This feature allows sections of the source code to be included, or omitted, when the program is assembled. For example, you may wish to include some debugging statements in your program, but assemble them only if a flag is set at the start of the source code. Conditional assembly usually takes the form of an IF...THEN...ELSE...ENDIF structure.

### REPETITIVE ASSEMBLY

This feature allows a section of the source code to be assembled a finite number of times. This can be very useful in creating data tables where each entry is related.

### MACROS

A Macro is a section of source code which only appears in the source listing once, but

which is used a number of times when the program is being assembled. For example, you might find that a particular pair of instructions appear together at numerous points throughout the program. Rather than typing these instructions in each time they appear, they can be used as the basis of a *macro definition*, and given a name which thereafter can be used to refer to those instructions. Each time the name of the macro appears as an instruction in the source code, it will be replaced at assembly by the individual instructions making up the macro.

Macro definitions can also include parameters which are specified each time the macro is used (similar to the arguments of a procedure). In this way macros can be used to create *pseudo-instructions* which can then be used normally in the program.

## LITERAL AREAS

One limitation of the ARM is its inability to include a wide range of immediate operands in instructions. This is because of the limited number of bits free in the instruction field. To get around this, constants can be stored in memory and loaded using program-relative addressing. Some assemblers automatically handle this, allowing the user to pretend that a real immediate constant was used.

## DATA AREA LAYOUT

A nice feature of an assembler is the ability to lay out an area of memory to be used as workspace automatically. This is done by defining the start address of the area and then using a special operator to allocate a particular sized chunk from the area.

## ACORN OBJECT FORMAT

Acorn object format (AOF) is effectively a way of specifying a machine code program which is not quite executable. This is because a program in AOF format may contain calls to routines which are not defined within the program, and in general the address at which the program will be loaded is undefined. To convert an AOF program into an executable one, the Acorn *Linker* is used. This basically takes several AOF programs and ties them together, making sure



## WHICH ASSEMBLER?

that all the routines called are defined somewhere within one of the programs. It then saves the final executable program. All the Acorn language compilers produce AOF output, and any assembly language program to be linked into a compiled program must therefore also be an AOF file.

### THE ACORN ASSEMBLER

This was the first Archimedes assembler available, and is (and will always be) the defacto standard against which others are judged. The Acorn assembler is supplied in two versions, one to produce stand-alone machine code programs, and the other to produce AOF files. Both versions are very similar, and both offer a very large number of functions. It would take too much space to discuss all the individual functions, just suffice to say that all the features mentioned above are provided.

Product:	Archimedes Assembler
Supplier:	Acorn Computers Ltd., Fulbourn Road, Cherry Hinton, Cambridge CB1 4JN. Tel. (0223) 245200
Price:	£226.37 (inc. VAT)

The documentation supplied with the Acorn assembler is extensive. It consists of a 177 page manual in the same format as that supplied with their C compiler. Not only is the assembler itself described, but also the ARM processor, including the instruction set of the Floating Point Emulator and the co-processor instructions. Unfortunately, the disc contains no sample programs.

### THE WINGPASS MACRO ASSEMBLER

This is the first alternative to the Acorn system, and has been designed with one particular application in mind - namely combining C programs with assembly language routines. This system is somewhat more limited than the Acorn one, for example it does not provide the automatic data area facility or the handling of literal areas. It does however provide a number of features that are solely related to C. For example, it is possible to include a C header file within the source code, and then refer to type definitions within the assembly language program. This makes the package ideal for writing functions

which could not be implemented in C, such as routines which must interact directly with hardware. This assembler can be used independently of C, but in this case a copy of Acorn's linker will be needed. This is supplied with all the languages, the Acorn assembler, and the Software Developer's Toolbox.

Product:	Macro Assembler
Supplier:	Wingpass Ltd., 19 Lincoln Avenue, Twickenham, Middlesex TW2 6NH.
Price:	£49.95 (inc. VAT)

The manual supplied with the Wingpass assembler is 72 pages long and contains less material than the Acorn one, partly because of the fewer features offered. However, the disc contains a number of example programs which illustrate all the available features.

### CONCLUSION

When comparing these two assemblers one has to consider not only the products themselves, but also their prices and the functions they aim to perform. There is no doubt that the Acorn system offers the most features, and is the most flexible. However, this is exactly what it is designed to do, and used in conjunction with the *Software Developer's Toolbox* (see RISC User Volume 2 Issue 3) provides a complete environment for developing assembly language programs. The price of this flexibility is however a massive £226.

On the other hand, the Wingpass assembler, while appearing similar at first sight, sets out to perform a totally different function. It is designed primarily for incorporating assembly language routines into C programs, and hence does for C what Basic's assembler does for Basic. The Wingpass assembler doesn't possess, and doesn't need, a number of the functions of the Acorn system. However, at £50, it is less than a quarter of the price of the Acorn system, and this is a saving which should not be ignored.

To summarise, both systems are well written and perform their tasks perfectly. However, before considering either you should think carefully as to whether Basic's assembler would be a suitable, and possibly better, alternative.

**RU**

# ARCHIMEDES SOFTWARE

(NOW RISC OS compatible)

## SYSTEM DELTAPLUS (v2) £79.95

This Database Management System is the ultimate in simplicity and flexibility. Up to 2 billion records. Multiple files open at any time. Multi-tasking allowing one file to be edited whilst searching another. Search by simple selection from within windows. Comprehensive maths, searches and sorts. Mailing labels, reports, card copier.

## ATELIER £99.95

(To be released soon). Professional quality art package. Functions include anti-alias squashing, rotation of areas into any quadrangle, and sprite and brush patterning. Further features allow TV-style techniques to even wrap items around 3D surfaces. (Multi-tasking if using RISC OS.)

## REPORTER £39.95

Fully relational user defined reporting package. Allows sub-totals and sub-sub-totals. Contains more facilities than dBASE III report writer! (This is an add-on utility to the System DeltaPlus database.)

## MAILSHOT £39.95

Versatile utility to produce mail-merged letters and labels across a sheet. (This is an add-on utility to the System Delta Plus database.)

## SIGMASHEET £69.95

Massive fast spreadsheet. Definable colours, spacing, widths, formats, etc. Over 30 maths/stats functions. Vertical/horizontal or automatic calculation modes.

## GAMMAPLOT £69.95

Extensive presentation graphics/charts with art package. Multiple charts/graphs on screen anywhere. Pens, shapes, patterns, fonts, block options, zoom, slide show facility. Imports pictures/logos. Merges with Wordprocessors.

Prices  
inclusive  
of VAT.

**MINERVA**  
69 SIDWELL STREET, EXETER. EX4 6PH  
Tel: (0392) 437756 Fax: (0392) 421762

Access &  
Barclaycard  
accepted



# ROBICO ADVENTURES FOR THE ARCHIMEDES

**RISE IN CRIME** (*Archimedes 305/310/400 £29.95*)

In the vast, galaxy-wide federation, you are one of the abnormal few: unconditioned, discontented and insubordinate! Can you rid yourself of the shackles of this conformist, mind-imprisoned society? Rise in Crime boasts over 396K of text, 400 individually described locations, 150 objects and several *high resolution colour graphic screens!* Mouse driven input and an advanced parser make this a superb addition to your Archimedes collection!

**FUGITIVE'S QUEST** (*Archimedes 310/400 £29.95*)

As Paul Preston, respectable schoolmaster, your life is shattered when you are convicted of murder – a crime you did not commit! Set in 1952, against the stunning backdrop of southwest England, the hangman's noose awaits unless you escape and find the evidence to clear your name! Fugitive's Quest is a large adventure with about 230 locations, 80 objects and reams of well written text. An advanced parser, mouse driven input and several *full colour, photo-quality illustrations* complete the unmistakable feel of Robico quality.

**ENTHAR SEVEN** (*Archimedes 305/310/400 £29.95*)

It is the distant future . . . Whilst on board a tiny Interplanetary Space Hopper its orbit begins to decay! Once on the planet surface can you escape and return, again, to the freedom of the stars? Enthar Seven is a disk based mega-adventure! It has 450 locations, 80 objects, a massive vocabulary and 1200 messages! And that's just the 8-bit versions! The Archimedes version has over 140K of text and some of the *best graphics you'll ever see!*

You can order by telephone using your ACCESS card, or by cutting out the coupon below. All orders are despatched first class by return.

To: ROBICO, 3 FAIRLAND CLOSE, LLANTRISANT, MID GLAMORGAN CF7 8QH

Tel: (0443) 227354

ADVENTURE	MODEL	QTY	PRICE	TOTAL
RISE IN CRIME	ARC		£29.95	
FUGITIVE'S QUEST	ARC		£29.95	
ENTHAR SEVEN	ARC		£29.95	

Name .....

Address .....

I enclose a cheque/postal order payable to 'ROBICO' for £ .....

Signature .....

or please debit my ACCESS account:

Expiry Date .....





# THE RISC USER NOTEPAD (2)

David Spencer continues this multi-tasking RISC OS application.

This month we will add the routines to perform text handling and screen update to the Notepad. Start by adding this month's listing to the !RunImage program from last month, making sure that the original has not been renumbered at all. You will find there are still gaps in the line numbering, and these are for the addition of the final part next month. You should also delete line 570 which was included in last month's listing. This line is not in fact needed until next month, and at the moment causes spurious errors when starting new applications from the Desktop.

## USING THE NOTEPAD

By now, the Notepad is usable to the extent that it can function as a jotter. It is not yet possible to load or save the contents, or to print them out.

When you open the Notepad window you will see the work area with a line across the top and two arrows and the page number above this line. The border is a pale yellow to show that the window will accept key presses, and a red caret can be seen in the top left hand corner. There are eight pages in all, each with fifteen lines of forty characters. You can move backwards and forwards through the pages by clicking on the arrows, and move the caret around the page with the cursor keys, or by moving the pointer and clicking.

There are two modes of text entry - Insert and Overtyping. Pressing the menu button while over the Notepad window will bring up a menu, the first two options of which allow you to change entry mode. The current mode is shown by a tick, the default being Insert. As the names suggest, overtype mode causes existing characters to be overwritten, while insert mode pushes characters to the right to fit the new one in. The text is totally free format, and inserting a character will cause all the text beyond it to be shuffled. As text is entered the caret is moved automatically, and running off the end of one page will cause the next to be displayed. The Return key will move the caret to the start of the next line without entering any character into the text.

The Delete key will delete the character to the left of the caret. In overtype mode the character

is replaced with a space, while in insert mode the gap is closed up by moving the remaining text back. The Copy key will delete the character to the right of the caret, and will always close the gap regardless of the entry mode. Finally, the Insert key will insert a space to the right of the caret. This is useful for adding one or two characters without having to enter insert mode.

As already said, the menu button will bring up a menu. The first two options control the entry mode as already explained. The third option will wipe all the pages of the notepad clean and, as no warning is given, should be used with care. The fourth and fifth entries are 'Print' and 'Save' respectively, and these will be implemented next month.

## HOW IT WORKS

The basic structure of the program is very similar to that of a normal WIMP based program, although there are a few obvious differences. If you are not familiar with a single-tasking WIMP program then it is perhaps worth reading the series in RISC User Volume 1 Issues 5 to 7. The best way to understand the listing is to work through the program with reference to both the *Programmer's Reference Manual* and the notes below.

The first point to note is the way in which SYS "Wimp\_Initialise" is called in line 170. This is called with three parameters. The first is the version number of the oldest Window Manager with which this program will work. This should be left as 200 in all programs. The second parameter is a four byte value made up of the characters 'TASK'. This tells the Window Manager that this program is multi-tasking. The final parameter is a text string which will be used by the Task Manager when it displays the table of memory usage. This should be kept short, but should give a clear indication of the function of the program. Finally, the SYS call returns a value which the program assigns to the variable *task*. This is the task handle, and is a unique value which the Window Manager can use to identify our task.

The next important stage is the creation of a user sprite area in lines 210 to 240. This is



essential when using sprites in a multi-tasking program, for the reasons discussed last month. The program 'peeps' at the sprite file before loading it so that the area set up can be exactly the right length. You will notice the use of the operating system variable `<Notepad$Dir>` in the filenames. This is set to the application directory pathname by the `!Run` file, and using it ensures that the program will be able to find the sprites regardless of the currently selected directory.

The two RND functions in line 250 randomise where on the screen the Notepad window will appear when it is first opened. This is so that if the program is installed several times, and each incarnation opened one after the other, all the windows will not stack exactly on top of each other.

The next section of code (lines 260 to 340) sets up a number of icons. The icon worthy of mention is that defined in line 280 where the icon is placed in window -1. This in fact refers to the right-hand side of the iconbar, and it is this statement which places the Notepad icon on the iconbar.

The all-important polling loop comes next. Again, this is very similar to that of a single-tasking program. The value passed to `Wimp_Poll` is set to prevent null reason codes being returned. Claiming these would slow down all the other active tasks. The other major change is the recognition of the new reason codes 17 and 18 in line 470. These are used to signal to our program that another task is sending us a *message*. Messages are effectively just blocks of data, and we will cover them in detail next month. For now, all that we need to know is that when a message is received, the word at offset sixteen in the returned parameter block indicates what purpose the message serves. The only value we are interested in at the moment is zero, which is a message sent by the task manager to tell us to shut down. This will occur either when the user kills our program from the task display, or when the Desktop is about to shut down. Line 550 of the program sets our 'Quit' flag when this message is received. To demonstrate the effect of the message, remove this line and try exiting from the Desktop.

The next section of the program is relatively standard. One point to note is that whenever the Window Manager refers to our icon on the iconbar, it uses a window handle of -2 to represent the iconbar. This can be seen in line 930 where button presses are detected, and line 1670 when the menus are popped up. Other things to note are the dimensions of the menu entries in `PROCdrawmenu`, and the positioning of the 'Quit' menu in line 1680. These values should be used in all multi-tasking programs so that menus appear uniform from program to program.

Lines 2170 to 2220 constitute the error trapping routine. This starts by cancelling any active drag box, setting a parameter block to contain the error information and then calling the routine `SYS "Wimp_ReportError"`. This displays an error box and returns when the user clicks on the OK button. At this point our program jumps back into the polling loop.

Most of the rest of the program is concerned with the text handling. You should note in particular the key processing routine (`PROCKey`). Any keys which are not recognised must be passed back to the Window Manager using `SYS "Wimp_ProcessKey"`. This is so that other tasks which respond to particular keys can react properly. If you remove line 2510, then pressing F12 to bring up a star prompt will no longer work if the Notepad has the input focus.

Finally, `PROCretile` closes a window and immediately reopens it. The purpose of this is to update the window title if it has been changed (such as a "" being added when the text is modified).

*Next month we will add the file handling and print routines, and conclude with an explanation of some of the more subtle points of the program.*

```

160 $fi="Notes":mod=FALSE
190 PROCwipe:insert=TRUE
200 PROCassemble
340 void=FNtexticon(main,280,-36,376,-
4,&7000119,pno,-1,7)
450 WHEN 8:PROCKey(block)
820 PROCworkarea
980 open=TRUE:PROCmovecaret
1000 WHEN main
1010 CASE b!16 OF

```

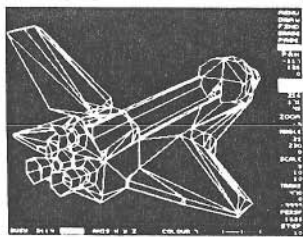


```
1020 WHEN !1:PROCputcaret(b)
1030 WHEN left:IF NOT FNbpage VDU 7
1040 WHEN right:IF NOT FNfpage VDU 7
1050 ENDCASE
1350 :
1360 DEF PROCworkarea
1370 ox=block!4-block!20
1380 oy=block!16-block!24
1390 MOVE ox,oy-40:DRAW BY 656,0
1400 ptr=data+page*600:oy-=56:ox+=8
1410 FOR line=0 TO 14
1420 IF NOT(block!40<oy-28 OR block!32>
oy) THEN MOVE ox,oy:SYS "OS_WriteN",ptr,
40
1430 oy-=32:ptr+=40
1440 NEXT
1450 ENDPROC
1520 DEF FNTtexticon(window,x0,y0,x1,y1,
flags,tptr,vptr,len)
1530 block!24=tptr:block!28=vptr
1540 block!32=len
1550 =FNiconblk(window,x0,y0,x1,y1,flag
s)
1560 :
1690 WHEN main:PROCdrawmenu(2,"Note Pad
,Insert,Over,Clear,Print,Save")
1700 IF insert THEN
1720 menu!32=menu!32 OR 1
1730 ELSE
1750 menu!56=menu!56 OR 1
1760 ENDIF
1780 SYS "Wimp_CreateMenu",,menu+4,!b-3
2,b!4+16
2070 WHEN 2
2080 CASE !b OF
2090 WHEN 0:insert=TRUE
2100 WHEN 1:insert=FALSE
2110 WHEN 2:PROCclear
2120 WHEN 3:PROCprint
2130 ENDCASE
2230 :
2240 DEF PROCputcaret(b)
2250 x=!b:y=b!4:!block=main
2260 SYS "Wimp_GetWindowState",,block
2270 x=x-block!4+block!20-8
2280 y=y-block!16+block!24+56
2290 IF y>=16 THEN ENDPROC
2300 IF y>0 THEN y=0
2310 IF y<-476 THEN y=-476
2320 IF x<0 THEN x=0
2330 IF x>632 THEN x=632
2340 x=(x AND NOT15)-16*((x AND 15)>7)
2350 tx=x DIV 16:ty=-y DIV 32
2360 IF tx=40 AND ty<>14 THEN tx=0:ty+=
1
2370 IF tx=40 AND ty=14 THEN tx=39
2380 PROCmovecaret
2390 ENDPROC
2400 :
2410 DEF PROCkey(b)
2420 IF !b=main THEN
2430 key=b!24
2440 CASE TRUE OF
2450 WHEN key=&18B:PROCdelete(TRUE)
2460 WHEN key=&7F:PROCdelete(FALSE)
2470 WHEN key=&D:PROCnewline
2480 WHEN key=&1CD:PROCinsert
2490 WHEN key<&100 AND key>&1F:PROCchar
(key)
2500 WHEN key>=&18C AND key<=&18F:PROCC
ursor(key-&18C)
2510 OTHERWISE SYS "Wimp_ProcessKey",ke
y
2520 ENDCASE
2530 ENDIF
2540 ENDPROC
2550 :
2560 DEF PROCcursor(dir)
2570 CASE dir OF
2580 WHEN 0:tx-=1
2590 WHEN 1:tx+=1
2600 WHEN 2:ty+=1
2610 WHEN 3:ty-=1
2620 ENDCASE
2630 IF tx<0 THEN tx=39:ty-=1
2640 IF tx>39 THEN tx=0:ty+=1
2650 IF ty<0 THEN ty=-14*FNbpage
2660 IF ty>14 THEN ty=-14*NOT FNfpage
2670 PROCmovecaret
2680 ENDPROC
2690 :
2700 DEF FNbpage
2710 IF page=0 THEN =FALSE
2720 page-=1:$pno="Page "+STR$(page+1)
2730 SYS "Wimp_ForceRedraw",main,0,-544
,656,0
2740 =TRUE
2750 :
2760 DEF FNfpage
2770 IF page=maxpage THEN =FALSE
2780 page+=1:$pno="Page "+STR$(page+1)
2790 SYS "Wimp_ForceRedraw",main,0,-544
,656,0
2800 =TRUE
```

# SILICON VISION

SOFTWARE FOR THE ARCHIMEDES & BBC

## SolidCAD



The ultimate 3D Draughting System for Architectural design, Interior design, Engineering Design and Teaching CDT. Allows drawing in plan, front & side elevations and also directly in 3D view. Includes powerful zoom & pan options for precision draughting and surface definition for creating solid colour objects. Also includes Sweep, Extrude & Macro facilities for designing very complex objects easily. Designs created with SolidCAD are compatible with the Realtime Graphics Language for high-speed flicker-free animation. The custom Archimedes version also performs smooth shading for realism. SolidCAD/Arc users can upgrade to the Realtime Solids Modeller (Arc) for £40.00.

£49.95 (ARC & BBC) New

## REALTIME SOLIDS MODELLER

The package includes both the sophisticated design environment of SolidCAD and the high-speed animation capability of a Realtime Graphics Language (RGL) module developed in pure ARM Risc code for supercharged performance. The package is ideal for Architectural design, Interior design, Engineering design & teaching CDT. The RGL module can be used to create standalone flicker-free animation of designs from your own programs. Smooth shading is also performed for realistic images. Through our in-house expertise in 3D Design and High speed techniques, no other package can rival the design speed, realism & animation speed of the Realtime Solids Modeller.

£89.95 (ARC) New

## REALTIME GRAPHICS LANGUAGE

The Realtime Graphics Language (RGL) provides a complete 3D Solids Wireframe & Shaded environment with 64-bit commands and 3D Editors for designing objects to animate them. Software programs include a 35,000 pixels/sec line generator for fast 3D drawing software. Archimedes, Scale-Draw, Perspective and Turtlegraphics. Also compatible with the RGL module with SolidCAD/BBC.

£49.95 (BBC)

## SUPER-DUMP

Block Dumping Utility, which takes advantage of the highest resolution capability of drawing & laser compatible printers to provide 1920 x 1024 resolution. Images can also be stored & published and previewed before printing. Fully compatible with SolidCAD, Realtime Graphics Language, Gate Array design system & 3D CAD Animation system. Images with graphics programs or other CAD packages can be made compatible with Super-Dump by the addition of a few simple commands. An example program is included in the package.

£15.95 (BBC), £24.95 (ARC) New

## Presentation Manager

Powerful presentation software which allows you to create, edit and play back computer presentations. Includes a wide range of animation and manipulation facilities. Also handles graph plotting for spreadsheets, databases, etc. Images which can be incorporated within the presentation.

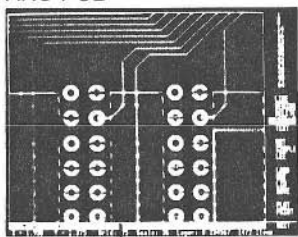
£34.95 (BBC), £49.95 (ARC) New

SILICON VISION LTD, SIGNAL HOUSE, LYON ROAD, HARROW  
MIDDLESEX HA1 2AG. TEL: 01-422 2274 or 01-861 2173  
FAX: 01-427 5169. TELEX: 918266 SIGNAL G.

☐ (Access/Mastercard/Eurocard accepted)

All prices include VAT and Carriage (Overseas orders add £4).

## ARC-PCB



The ultimate PCB design system developed specifically for the Archimedes with a specification that cannot be matched. Includes Automatic routing, Rats-nesting, 8 layers, Surface mount capability, 0.001 resolution, 32 x 32 maximum board size, On-line Help, Fast Zoom/Pan/Redraw, Text & Silkscreen facility, Variable Line/Pad/Text/Grid sizes, Part Libraries, Block Move/Copy/Rotate/Mirror/Erase options, and up to 300,000 components. For hardcopy, the system supports a large number of plotters and ordinary Epson compatible printers at very high resolutions (1920 x 1024) of 240 dots/inch for near laser quality output. An entry-level ARC-PCB system (without auto-routing) is available for £99.95 (Auto-routing upgrade: £100). Enquire about our turnkey PCB design systems complete with Colour Archimedes and/or plotters with prices from £1220 to £3909 (inc VAT).

£195.00 (ARC) New

## RiscBASIC

Supercharge your Archimedes Basic programs by compiling them automatically into pure ARM Risc code with the RiscBASIC compiler. Features include Relocatable modules, Cross reference of all variables, functions and procedures, Floating point and Integer support, Stand alone code generator. Optimising compiler & Full runtime error handler.

£99.95 (ARC) New

## RiscFORTH

A new 32-bit implementation of the FORTH 83 standard, designed to take full advantage of the ARM architecture. Features include Multi-tasking, Optimising compiler, built in ARM assembler with floating point instructions, built in Full screen Editor, File system interface, OS calls support, Floating point & Integer maths, WIMP support, Single step debugger, Shadow screen for documentation, Block manipulation, Dictionary & Vocabulary display, Cross finding and a standalone code generator.

£99.95 (ARC) New



```
2810 :
2820 DEF PROCmovecaret
2830 SYS "Wimp_SetCaretPosition",main,-
1,tx*16+8,-ty*32-88,40+(1<<24),-1
2840 ENDPROC
2850 :
2860 DEF PROCwipe
2870 FOR ptr=0 TO (maxpage+1)*600-1 STE
P 4
2880 data!ptr=&20202020
2890 NEXT:tx=0:ty=0
2900 page=0:$pno="Page "+STR$(page+1)
2910 ENDPROC
2920 :
2930 DEF PROCclear
2940 PROCwipe
2950 PROCretitle(main)
2960 SYS "Wimp_ForceRedraw",main,0,-544
,656,0
2970 ENDPROC
2980 :
2990 DEF PROCchar(key)
3000 ptr=page*600+ty*40+tx
3010 IF insert THEN
3020 IF data?4799=ASC" " THEN
3030 PROCshunt(ptr,TRUE)
3040 data?ptr=key
3050 SYS "Wimp_ForceRedraw",main,0,-544
,656,-(52+32*ty)
3060 ELSE VDU7
3070 ENDIF
3080 ELSE
3090 data?ptr=key
3100 SYS "Wimp_ForceRedraw",main,8+16*t
x,-(87+32*ty),24+16*tx,-(52+32*ty)
3110 ENDIF
3120 PROCmodified
3130 PROCcursor(1)
3140 ENDPROC
3150 :
3160 DEF PROCshunt(from,dir)
3170 A%=data+from
3180 B%=data+(maxpage+1)*600-1
3190 IF dir THEN C%=1 ELSE C%=-1
3200 CALL code
3210 ENDPROC
3220 :
3230 DEF PROCmodified
3240 IF NOT mod THEN
3250 mod=TRUE:$mt+=" *"
3260 PROCretitle(main)
3270 ENDIF
3280 ENDPROC
3290 :
3300 DEF PROCinsert
3310 LOCAL insert:insert=TRUE
3320 PROCchar(32):PROCcursor(0)
3330 ENDPROC
3340 :
3350 DEF PROCdelete(dir)
3360 IF NOT(insert OR dir) THEN
3370 PROCcursor(0):PROCchar(32)
3380 PROCcursor(0)
3390 ELSE
3400 ptr=page*600+ty*40+tx+(dir=FALSE)
3410 PROCshunt(ptr,FALSE)
3420 data?((maxpage+1)*600-1)=ASC" "
3430 SYS "Wimp_ForceRedraw",main,0,-544
,656,-(52+32*ty)
3440 IF NOT dir THEN PROCcursor(0)
3450 IF NOT dir AND tx=39 THEN SYS "Wim
p_ForceRedraw",main,8+16*tx,-(87+32*ty),
24+16*tx,-(52+32*ty)
3460 ENDIF
3470 ENDPROC
3480 :
3490 DEF PROCnewline
3500 tx=0:PROCcursor(2)
3510 ENDPROC
3520 :
3530 DEF PROCassemble
3540 DIM code 200
3550 FOR pass=0 TO 2 STEP 2
3560 P%=code
3570 [OPT pass
3580 \Move (R0)...(R1) up/down 1 byte
3590 MOV$ R2,R2:BMI startlo
3600 .loop LDRB R2,[R1]
3610 STRB R2,[R1,#1]:SUB R1,R1,#1
3620 CMP R1,R0:BCS loop
3630 MOV PC,R14
3640 .startlo LDRB R2,[R0,#1]
3650 STRB R2,[R0]:ADD R0,R0,#1
3660 CMP R0,R1:BLS startlo
3670 MOV PC,R14
3680 ]NEXT
3690 ENDPROC
3700 :
3710 DEF PROCretitle(!block)
3720 SYS "Wimp_GetWindowState",,block
3730 SYS "Wimp_CloseWindow",,block
3740 SYS "Wimp_OpenWindow",,block
3750 PROCmovecaret:open=TRUE
3760 ENDPROC
```

# OS and BASIC V Guides

Archimedes Basic Compiler Version 2 • Archimedes Operating System Guide  
BASIC V Guide • Special Offers on Archimedes PC Emulator and ANSI C

## ABC Version 2

Don't live on promises, buy the only true BASIC V Compiler currently available, and now in its second release! But don't take our word here's what the reviewers said:

"...Excellent Dabs Press product. Buy it!" RISC User Dec.88

"ABC is a vital part of any programmer's toolbox, it puts compilers on other systems to shame. Unquestionably one of the most impressive pieces of software I have yet seen running on the Archimedes." A&B Dec 1988

"...I can tell you now, I am very impressed. This is a superb package." Archive Dec 1988

The above quotes were referring to Version 1 of ABC - ABC Version 2 is even better! Version 2 allows use of double and extended precision floating point, multiple exits from procedures and functions, RETURN parameter passing, new compiler directives and very much more.

As all reviewers have found, ABC makes writing machine code programs and relocatable modules easy. Programs can be written using the full range of BASIC V's error messages and report's, and once fully working, run through ABC which transforms them into machine code, ready for immediate action.

Demo Disc: We have produced a demo disc of ABC which still supports over 100 commands. It costs just £2 and this is refundable on any subsequent purchase. A full specification sheet is also available on request.

Free upgrades!

**£99.95**

**DABS  
PRESS**

## Archimedes OS Guide

Bought RISC OS? Struggling with Arthur? Then you need our Dabhand Guide to the Archimedes Operating System. This users guide explains how the OS works and shows you how to get the very best from it.

In 320 pages you are introduced to the RISC technology before getting down to the nitty-gritties. Areas covered include SWIs, The CLI, the filing system, the FileSwitch module, Modules, writing applications, the window and font managers, sound and voice generation and the floating point model.

As with all Dabhand Guides the book includes practical examples for you to type in any try for yourself. The accompanying programs disc contains over 40 programs including several not included in the book.

The price of the guide is just **£14.95** or **£21.95** including the programs disc and manual.

## Archimedes Games!

**Arcendum: Board Game Family Fun** Backgammon, Draughts, Reversi and Quadline - four games for the price of one! Sure to get the whole family using the Archimedes. Just **£14.95**.

### Alerion: Arcade Action!

The highly acclaimed all-action shoot-em-up for the Archimedes. 256 colour mode, with digitised speech. Impossible to finish! Superb fun! Price **£14.95**.

## C: A Dabhand Guide

PCW said: "I only wish this book had been available when I was learning C." If you want to learn ANSI C then this 512 page volume is the way to do it. At **£14.95** it represents quite incredible value. Book and programs disc **£21.95**.

## BASIC V: A Dabhand MiniGuide

For anybody interested in BBC BASIC then this book from **Beebug Editor Mike Williams** is essential reading. Assuming a familiarity with BBC BASIC the various many new components of BASIC V are fully described using example programs throughout.

an essential aid for all Archimedes owners, and including coverage of RISC OS BASIC. Price **£9.95** - 128 pages - available end of April.

## Archimedes PC Emulator Shareware

Five discs full of PC software tested with the emulator to ensure compatibility. The collection includes software you would normally expect to pay a fortune for and includes a wordprocessor, spreadsheet, games, flowchart designer, printer utilities, and more. Price **£34.95**.

## SPECIAL OFFERS!

**PC Emulator & FREE Shareware** Purchase the Archimedes PC Emulator from us and we'll give you our Shareware pack - detailed above and worth **£34.95** - absolutely free of charge. Price just **£113.85 inc.**

**ANSI C & FREE C Dabhand Guide** The best combination! Purchase Acornsofts ANSI C and we'll throw in our top-selling Dabhand Guide to C - 512 pages and worth **£14.95**. Price just **£113.85 inc VAT & p&p**.

**FREE on Request  
Bumper Catalogue!**

5 Victoria Lane (RUA), Whitefield, Manchester M25 6AL

Tel: 061-766 8423 BT Gold: 72:MAG11596 Prestel: 942876210

Prices include VAT and P&P (UK/BFPO/CI). ACCESS/VISA accepted by post/phone/mailbox/in person. Cheques and POs to address above. Dabs Press products available from all good dealers. Add £2.50 (£12 air) if outside UK. Official orders welcome.



# Postbag

*This month's Postbag concentrates on readers' views about RISC User.*

## READERS' COMMENTS ON RISC USER

In the December issue of RISC User (Volume 2 Issue 2) I noticed with interest a letter from Barbara Wilson pointing out that the magazine was too technically oriented. With this I agree. For every computer boffin among your readers there must be many others at various levels of knowledge down to absolute beginners.

With RISC User I would like to see many more short (or rather short) utility routines which could be incorporated into one's own programs. The Hints & Tips section I would like to see enlarged. In general I would like to see more space devoted to simpler matters.

**Arthur Cantrill**

Thanks for a great magazine in RISC User. I am currently experimenting with the PC Emulator, but find the manual a little short on instructions, especially where PC DOS commands are concerned. Would you please run an article on these commands?

**Bill Tottle**

As a new computer user (the Archimedes is our first), we are especially interested in articles which introduce us to the available software, any new software in the pipeline, and general articles to give us some idea of the wide range of things which can be performed on this computer. I am well aware that at the moment we are just scratching the surface.

In view of this, we find the majority of your articles almost incomprehensible, and would appreciate it if you could begin some articles with an "idiot's guide", to be skipped by those already in the know. Or perhaps you could have a section of the magazine especially for people like us with limited knowledge of programming.

**Pamela Price**

*Most of the recent correspondence, as above, has been from RISC User members seeking more introductory and explanatory articles in the magazine. We are already try-*

*ing to respond to the needs of these readers through a greater emphasis on applications, and by including more introductory articles on the Archimedes, but it should be remembered that this is a highly sophisticated, and thus sometimes complex system.*

*We would also welcome comment from our more technically minded readers (the boffins as Mr Cantrill phrases it). What do they think about RISC User content? We feel that we have an obligation, which we try to meet, of catering for the needs of all Archimedes users, and all comment from readers in this respect is welcome.*

## VIDEO CAMERAS

If you are looking for subject matter, may I make a plea on behalf of video camera owners like myself. So far the majority of software available for video titling has been written for the Amiga, and I have been waiting for someone to port it over to the Arc. Alas this has not happened.

I have used the splendid 'Fancy Font' scroller (Volume 1 Issue 7) written by Barry Christie. Good as it is, it is not that easy to use, especially for non-programmers. I wonder if the program could be enlarged and made more comprehensive. Additional items might be:

1. Menu driven
2. Four way scrolling, possibly 3D
3. Coloured titles (for an Arc with a PAL colour encoder module) and B/W titles
4. The two Acorn fancy fonts at different point sizes.
5. Extra fonts, fancy or otherwise
6. To be used with or without genlock
7. Wipe patterns and fades

It seems that I am asking a lot and by its complexity too long for the magazine, but if it were available as saleable software I am sure that there would be a ready market. I feel that there is about to be an explosion in the sale of video cameras, but unfortunately there is very little post production software, and none for the Archimedes.

**R.Follett**

*Can anyone help? This seems like a good opportunity for someone with the necessary time and expertise.*

**RU**

# PIPEDREAM

POWER AT YOUR FINGERTIPS

## PIPEDREAM

- o The best word processor for the Archimedes. It has dynamic on-screen page breaks, multi-column text and 'live' embedded numbers and calculations.
- o The fastest and most powerful spreadsheet for the Archimedes. It can sort, format text around numbers and store and quickly recalculate large models.
- o The most flexible database for the Archimedes. It can store a huge number of fields, search, sort and select records and have fields which are calculated from other values.
- o Complete integration of word processor, spreadsheet and database – no need to exchange data between several programs.
- o A mail-shot program which can store names and addresses, write the letter to send to them and selectively mail to each one, including calculations.
- o Ideal for writing invoices, statements and estimates: the text and figures can fit easily into a pre-designed form – and, of course, the figures are 'live'.
- o An excellent way to record customer or sales information: keep a database of sales leads, sort it on different criteria, produce the required totals and select and print out the key information.
- o Compatibility with 4 different computers: the BBC Micro, the Acorn Archimedes, the Z88 and the IBM PC.
- o A pull-down menu and help system to make it easy for you to get started.
- o Built-in file transfer with PipeDream on the portable Z88 so that you can continue work away from your main computer.


## SPELLCHECK

- o PipeDream SpellCheck is now available, with a dictionary of 93,003 words and the ability to check over 40,000 words per minute on the Archimedes.

## VIEW PROFESSIONAL

- o The BBC Micro version of PipeDream which provides complete file compatibility with PipeDream and with the Z88 and which shares many of PipeDream's features.

PipeDream costs only £99.00 plus VAT, View Professional £60.00 plus VAT and SpellCheck £43.00 plus VAT.

To order PipeDream, View Professional or SpellCheck, or to find out more, call us on 0954 211472. You can pay by VISA or Access by phoning us with your card number. 

colton  
software

FOR A FREE BROCHURE, COMPLETE AND SEND THIS COUPON

PipeDream ☐ View Professional ☐

RU3/89

Name

Address

Post Code

COLTON SOFTWARE, BROADWAY HOUSE,  
149-151 ST NEOTS ROAD, HARDWICK,  
CAMBRIDGE CB3 7QJ, ENGLAND

# TECHNICAL QUERIES

Our regular column dealing with readers' technical questions.

## HARD FACTS

Dear RISC User,

*There are now a number of hard disc upgrades for the 310, all appearing to offer similar features at varying prices. Has RISC User tested any of these? Also, what interface do these hard discs use? Is it SCSI?*

Kevin Long

RISC User has not as yet published a comparative review of hard disc upgrades, but we may do so in a future issue. As you say, most appear to offer very similar features and performance. The two main criteria that can be used as a comparison are the storage capacity of the drive, and the speed of access. This latter value is normally measured by timing a series of standard benchmark programs which perform various file operations. A less obvious difference between upgrades is the ability to support a second hard drive. For example, the Acorn upgrade can be extended simply by connecting an additional hard drive to the podule, while the Watford Electronics upgrade doesn't offer this feature. It does not follow, however, that the more you pay the more features you get, so you should study all the options before buying.

On your second point, the interface used by the hard drives is a standard interface called ST506. This is basically a hard disc equivalent of the interface used by all floppy disc drives. The podule supplied with the hard disc upgrade contains a disc controller which allows the computer to connect to the ST506 interface. SCSI (Small Computer Systems Interface) is another interface which is found on some hard drives. It offers the advantage that it defines a set of commands for accessing devices, as well as specifying how they are connected. This means that, for example, a hard drive could be replaced by a tape streamer, or even a CD ROM, and the software will still be able to access it totally unaware of the actual hardware present. We believe that a SCSI interface podule for the Archimedes will be released in the not too distant future, though no details can yet be given.

David Spencer

## MASTERING THE DISC MENU

Dear RISC User,

*I have a few questions about the RISC User ADFS Menu. I have been using this for some time now, but there are one or two things that I can't get it to do. Firstly it is infuriating that although it will run any file by double-clicking on it, you cannot just load a program without running it. Secondly it seems to hang with a read error message when clicking on certain file types; and thirdly I have problems making out some of the colours in the revised menu, and would prefer a mono screen - is there a way to achieve this.*

Arch Busby

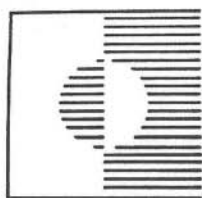
First of all there is a way to LOAD a program by double-clicking - just double-click with the right-hand button. You may also be interested to know that double-clicking with the middle button has a very special effect too. For details, see RISC User Volume 1 Issue 3.

We suspect that your 'hanging' errors are caused by clicking on files whose file types cause the file to be loaded into RAM at an unsuitable point, so corrupting the machine's stack or whatever. In fact the effect of double-clicking (with left or right-hand buttons) on any file is not determined by the ADFS Menu module at all, but by the aliases set up in your machine (either by default or by yourself). You can determine what happens to any type of file when you double click with the left hand button (i.e. run) or with the right hand button (i.e. load) by setting up run and load aliases. See RISC User Volume 2 Issue 4.

Altering the colours in the new version of the ADFS Menu is a relatively easy matter. A short article in RISC User Volume 2 Issue 1 explains how to do it. With a little experimentation, you should be able to render the whole menu in blacks, whites and greys.

Lee Calcraft

**RU**



# Lingenuity

Specialist Software

## Archimedes *PRODUCTS FOR THE 1990's*

### NEW PRESENTER STORY

*The ultimate in Business Presentation software. Turn your Archimedes into a tool to project combined images of Text, Logos, Graphs, Digitised & Video Pictures to illustrate your presentations more clearly. Many special screen features are included such as vertical and smooth scrolling and bouncing. It can be used with most makes of RGB projectors or you can use your Archimedes monitor for smaller Presentations. PRESENTER STORY is available for Business applications now. Education and Media/Video versions will be available soon. RISCOS compatible.*

**£199.00 + VAT**

### PRESENTER

*The established presentation package for Archimedes users in schools and businesses alike. Colour 3-D Bar-, Line- and Pie- charts and graphs can be made quickly and easily in the Acorn WIMP environment. These charts and graphs can be printed on FX-compatible and Integrex colour printers and also on Plotmate plotter. Alternatively, they can be ported into other packages, such as 1st Word Plus, Artisan & Graphic Writer. RISCOS compatible.*

**£24.95 + VAT**

### NEW CONTROL PANEL

*Confused about the \*Configure commands? With Control Panel there is no need to worry. Use Control Panel to set the right Archimedes' configuration for all your application software. You can then Save and Load the configuration settings of your choice for use at later dates. Control Panel also includes a utility to build configuration files into program boot files. Fully wimp-driven with constant real-time help screen.*

**£14.95 + VAT**

### NEW COLOUR CONVERTER

*Do you want to capture colour images onto your Archimedes screen? The Colour Converter module running with the Watford Digitiser allows full colour images to be captured and saved as screens or sprites. The package also includes new Dithering Software which will give you greatly enhanced near-television quality pictures, using an effective palette of 256,000 colours! The Colour Converter 1/2 Width Module and Software is available now.*

**£169.95 + VAT**

### NEW A I M (Archimedes Image Manager)

*Public Domain Image Processing software from Delft University which will increase the contrast and sharpen blurred images that have been digitised. Many varied uses—in education, industry and the home. A multisync monitor is required in order to give the necessary resolution.*

**£9.00 + VAT**



VISA  
and  
ACCESS

**All Lingenuity products are available direct or from good Acorn dealers.**

**Educational discounts and site-licences are available.**

**For further details please contact us on 098 685 477 or write to**

This month's collection of hints and tips is rounded up by David Spencer.

## DESKTOP SCROLLING

By P.D. White

When the RISC OS Desktop is started up, it enables the scroll protect option. This means that if a character is printed in the last column of a line, the cursor is not moved on to the next line until the next character is printed. This prevents the screen scrolling when a character is printed in the very bottom right-hand corner. However, this option is left on when the Desktop is quit, and it survives mode changes. While this will not normally cause problems, it can with some programs. To turn the scroll protect off use:

```
VDU 23,16|
```

## ALIGN AGAIN

By David Spencer

When Basic's built-in assembler aligns the instruction pointer to a word boundary, either implicitly or using the ALIGN directive, it does not pad the skipped bytes with any data. This means that if you assemble the same source program on two separate occasions, the two object codes will not necessarily be identical. This won't affect execution of the program in any way, but can produce confusing results if a bitwise comparison of two programs is performed.

## EXTENDED MOUSE

by Tom Short

The MOUSE statement in the new RISC OS Basic (Version 1.04) has been extended to provide a fourth value. By using the statement:

```
MOUSE x,y,b,t
```

*x* and *y* will contain the position of the mouse, and *b* the button state, just as before. The new variable *t* will contain a value called the *monotonic time*. This is a counter which starts at zero when the computer is first turned on, and is incremented one hundred times a second. It can only be reset by turning the computer off. The monotonic time can be used to time the delay between mouse clicks to detect double clicks, triple clicks etc.

## FINDING FILEINFO

by Ian Taylor

The format of the display produced by \*EX and \*INFO has been changed in RISC OS. In particular, they no longer

give the location on disc of a particular file, and for longer files they give the length as a rough number of kilobytes, rather than an actual value. To find out the full information use the command:

```
*FILEINFO <filename>
```

This will print out full file information for the named file, or in the case of a wildcarded name, a group of files. The file length (in hex) is the first value after the date.

## DISC DEFECTS

by John Regan

The new 'E' disc format offered by RISC OS allows discs to be used even if they have damaged sectors. When the disc is first formatted, it is verified, and if any sectors are found not to format, the disc is reformatted with those sectors mapped out. The mapped out sectors will appear as free space if the \*MAP command is used, and the disc can be used perfectly normally, albeit with a reduced capacity. However, it is not possible to use \*BACKUP to backup onto a disc containing defects.

## FILE OPENING

by Michael Thornton

Using OPENIN or OPENUP to open a file returns a file handle of zero if the file was not found. This handle must then be checked, and an error given if appropriate. However, if you bypass OPENIN and OPENUP, and instead use the call SYS "OS\_Find", you can force an error to be generated if the file is not found. To do this, use:

```
SYS "OS_Find",&44,name$ TO handle  
in place of:
```

```
handle = OPENIN name$
```

and:

```
SYS "OS_Find",&C4,name$ TO handle  
instead of:
```

```
handle = OPENUP name$
```

## DIRECTORY VIEWERS

by Graeme Davidson

The RISC OS Filer provides a nice feature to allow you to traverse a directory tree without ending up with countless directory viewers open. Double clicking on a directory in a viewer using the Adjust (right-hand) button, will open the

# HINTS & TIPS

# HINTS & TIPS

new directory, but also close the current viewer. Similarly, closing a directory viewer with the Adjust button will re-open its parent if it is not already visible. Running an application with a double-click of Adjust will close the viewer before the application is run.

## THE SHARED C LIBRARY

by David Spencer

The new release 2 of the Archimedes C compiler supports a shared library facility. This allows several C programs running simultaneously (multi-tasking under RISC OS) to share a common set of library routines, so reducing the length of the individual programs. In order to use this feature, the additional parameter "-l arm.clib.o.stubs" should be added to the command line when the source code is compiled. So, for example:

```
cc TestProg
```

would become:

```
cc TestProg -l arm.clib.o.stubs
```

The additional parameter tells the linker to include the object file 'stubs', rather than the standard library files.

When linked in this way, the program will only run with the shared library installed (otherwise the error 'SWI &80060 not known' will result). The shared library module has the filename 'Clib' and can be found in the library directory of the C compiler disc, or in the Modules directory within the /System application on RISC OS applications disc number 1.

## SYS FLAGS

by Glynn Clements

Most people who have used the SYS statement from Basic will know that the contents of registers can be returned by using the keyword TO followed by a comma-separated list of variable names. It is also possible to read the processor flags returned by the call, by placing a variable name preceded by a semicolon after the returned variables. For example:

```
SYS "OS_ReadC" TO key ;flags
```

The flags are in the form of a four-bit number made up of the flags N, Z, C and V. Each flag can be tested individually using AND. For example:

```
IF (flags AND 2) THEN PRINT "Carry Set" RU
```

## Archimedes Macro Assembler Version 2

The Wingpass Archimedes Macro Assembler takes ARM assembly language source files and produces Acorn Object Format (AOF) files suitable for linking with other AOF files produced by a high level language compiler.

### Features:

- \* Accepts the full ARM instruction set including Floating Point instructions. All FP data types (Float, Double, Extended and Packed) are supported.
- \* Allows 'C' language header files to be included so definitions of structures, unions and #defines need only be done once.
- \* Powerful macro facility includes parameter specification by position or by name, parameter defaults and variable length parameter lists.
- \* Example files supplied include: routines calling and called from 'C' routines obeying the calling standard, example relocatable module, complete Mandelbrot set program.

**£ 49.95** (inc. VAT and P&P)

Wingpass Ltd, 19 Lincoln Ave, Twickenham, MIDDLESEX TW2 6NH

**BEEBUG Ltd**  
Dolphin Place, Holywell Hill, St. Albans,  
Herts AL1 1EX  
Tel. (0727) 40303

# RISC USER magazine

## MEMBERSHIP

RISC User is available only on subscription at the rates shown below. Full subscribers to RISC User may also take out a reduced rate subscription to BEEBUG (the magazine for the BBC micro and Master series).

*All subscriptions, including overseas, should be in pounds sterling. We will also accept payment by Connect, Access and Visa, and official UK orders are welcome.*

### RISC USER SUBSCRIPTION RATES

£14.50  
£20.00  
£25.00  
£27.00  
£29.00

1 year (10 issues) UK, BFPO, Ch.I  
Rest of Europe & Eire  
Middle East  
Americas & Africa  
Elsewhere

### RISC USER & BEEBUG

£23.00  
£33.00  
£40.00  
£44.00  
£48.00

### BACK ISSUES

*We intend to maintain stocks of back issues. New subscribers can therefore obtain earlier copies to provide a complete set from Vol.1 Issue 1. Back issues cost £1.20 each. You should also include postage as shown:*

#### Destination

First Issue  
Each subsequent Issue

#### UK, BFPO, Ch.Is

60p  
30p

#### Europe plus Eire

£1  
50p

#### Elsewhere

£2  
£1

## MAGAZINE DISC

*The programs from each issue of RISC User are available on a monthly 3.5" disc. This will be available to order, or you may take out a subscription to ensure that the disc arrives at the same time as the magazine. The first issue (with six programs and animated graphics demo) is at the special low price of £3.75. The disc for each issue contains all the programs from the magazine, together with a number of additional items by way of demonstration, all at the standard rate of £4.75.*

### MAGAZINE DISC PRICES

Single issue discs  
Six months subscription  
Twelve months subscription

#### UK

£ 4.75  
£25.50  
£50.00

#### Overseas

£ 4.75  
£30.00  
£56.00

*Disc subscriptions include postage, but you should add 60p per disc for individual orders (30p for additional discs on the same order).*

*All orders, subscriptions and other correspondence should be addressed to:*

**RISC User, Dolphin Place, Holywell Hill, St Albans, Herts AL1 1EX.**

**Telephone: St Albans (0727) 40303**

*(24hrs answerphone service for payment by Connect, Access or Visa card)*